ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

КОМПЬЮТЕРНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Методические рекомендации к лабораторным работам для студентов направления подготовки 09.03.01 «Информатика и вычислительная техника» дневной формы обучения

Могилев 2018

УДК 004
ББК 32.973
К 63

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Автоматизированные системы управления» «31» августа 2018 г., протокол № 1

Составитель канд. техн. наук, доц. В. П. Василевский

Рецензент канд. техн. наук, доц. И. В. Лесковец

Методические рекомендации к лабораторным работам предназначены для студентов направления подготовки 09.03.01 «Информатика и вычислительная техника» дневной формы обучения.

Учебно-методическое издание

КОМПЬЮТЕРНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Ответственный за выпуск	А. И. Якимов
Технический редактор	А. А. Подошевко
Компьютерная верстка	Е. С. Лустенкова

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

Издатель и полиграфическое исполнение:
Государственное учреждение высшего профессионального образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1 /156 от 24.01.2014.
Пр. Мира, 43, 212000, Могилев.

Содержание

Введение

Цель дисциплины – получение знаний и навыков использования актуальных, на взгляд автора, для студентов направления подготовки «Информатика и вычислительная техника» компьютерных информационных технологий.

Лабораторный практикум содержит 10 лабораторных работ, выполнение которых закрепляет материал, изложенный в курсе лекций «Компьютерные информационные технологии».

Из обширного перечня применяемых компьютерных информационных технологий в данном курсе изучаются следующие системы:

– Microsoft Access, лидирующая среди систем управления базами данных (СУБД) для настольных систем, функционирующих под управлением ОС Windows;

– матричная система Matlab, занимающая лидирующее место в области научно-технических вычислений, расчетов и моделирования.

Для защиты лабораторных работ необходимо представить преподавателю файлы с результатами выполненных заданий, а также ответить на вопросы по тематике защищаемой работы.

Созданные при выполнении заданий файлы сохраняются в рабочей папке студента.

Основной формой защиты лабораторных работ является *собеседование.*

1 Базовые сведения о СУБД Access

1.1 Лабораторная работа № 1. Создание и сохранение базы данных заданной структуры

Цель: создать однотабличную базу данных (БД), освоить формирование структуры и ввод данных в режиме таблицы и с использованием формы.

Теоретические сведения

Основная функция Access, как и любой другой реляционной СУБД, – это работа со структурированной в виде таблиц информацией. Данные организованы в таблицах таким образом, чтобы обеспечить объединение разнородной информации, исключить ее дублирование, а также сделать доступными имеющиеся сведения.

Реляционные СУБД используют модель данных, предложенную в 1970 г. Э. Ф. Коддом, который показал, что набор двумерных таблиц при соблюдении определенных ограничений применяют для хранения данных.

В терминологии Кодда такие таблицы называются *отношениями (*англ. *relation*), отсюда и название баз данных – реляционные.

Программа Access позволяет обеспечить ввод данных в таблицы базы данных (БД), их хранение и сопровождение, а также получать из совокупности этой информации нужные данные. Применительно к приложению MS Access БД это совокупность таких объектов, как таблицы (основа БД), запросы, формы, отчеты, макросы и модули.

С понятием БД связано понятие СУБД. Это комплекс программных средств, предназначенных для создания структуры новой базы, наполнения ее данными, редактирования и визуализации информации.

Создание новой базы данных

После запуска приложения MS Access на странице **Приступая к работе** с MS Office Access в разделе **Новая пустая база данных** выберите команду **Новая база данных.**

В области **Новая база данных** в поле **Имя файла** введите имя файла. Если имя файла указано без расширения, расширение будет добавлено автоматически (*.accdb). Чтобы сохранить файл в другой папке, отличной от используемой по умолчанию, нажмите кнопку **Открыть** (рядом с полем **Имя файла**), перейдите к нужной папке (папку создать предварительно) и нажмите кнопку **ОК**.

Имя базы данных задайте **Деканат**, а тип файла оставьте прежним, так как другие типы файлов нужны в специальных случаях.

Нажмите кнопку **Создать.**

Приложение **Access** создаст базу данных с пустой таблицей с именем **Таблица 1** и откроет эту таблицу в режиме таблицы. Курсор находится в первой

пустой ячейке столбца **Добавить поле**. Как правило, для базы данных требуется создавать таблицы, характеристики которых отличаются от установленных по умолчанию. В подобных случаях используется конструктор таблиц.

Создание структуры таблицы базы данных

При создании структуры режим конструктора является функционально наиболее полным. С использованием конструктора устанавливаются имена полей, типы и свойства данных. Каждому типу данных соответствует определенный набор свойств.

Режим используем при создании структуры в соответствии с данными таблицы 1.1.

Затем можно переключиться в режим таблицы для ввода данных или ввести данные, применяя другой метод, например, вставку или импорт.

Таблица 1.1

Имя поля	Тип данных	Размер поля
Код преподавателя	Счетчик	
Фамилия	Текстовый	15
Имя	Текстовый	15
Отчество	Текстовый	15
Должность	Текстовый	9
Дисциплина	Текстовый	11

Выполните следующее.

1 На вкладке **Создание** в группе **Таблицы** щелкните **Конструктор таблиц**.

2 Для каждого поля в таблице введите имя в столбце **Имя поля**, а затем в списке **Тип данных** выберите тип данных.

3 Поля вкладки **Общие** оставьте установленными по умолчанию.

4 Для сохранения таблицы:

– выберите пункт меню **Файл, Сохранить**;

– в диалоговом окне **Сохранение** введите имя таблицы **Преподаватели**;

– щелкните по кнопке **ОК**.

5 Введите ограничения на данные, вводимые в поле **Должность**: должны вводиться только слова **Профессор**, **Доцент** или **Ассистент**.

6 Задайте текст сообщения об ошибке, который будет появляться на экране при вводе неправильных данных в поле **Должность**.

7 Задайте значение по умолчанию для поля **Должность** в виде слова **Доцент**.

8 Введите ограничения на данные в поле **Код преподавателя** – эти данные не должны повторяться.

9 Создайте столбец подстановок для поля **Дисциплина**.

10 Для задания условия на значение для вводимых данных:

– войдите в режим **Конструктор** для проектируемой таблицы. Если вы находитесь в окне базы данных, то выберите вкладку **Главная** и щелкните по группе команд **Режим**. Если вы находитесь в режиме таблицы, то щелкните по кнопке **Конструктор**;

– в верхней части окна щелкните по полю **Должность**;

– в нижней части окна щелкните по строке параметра **Условие на значение**;

– щелкните по кнопке для определения условий на значение при помощи построителя выражений;

– в появившемся окне напишите слово **Профессор,** затем выберите функцию ИЛИ, напишите **Доцент**, снова выберите или напишите **Ассистент** и щелкните по кнопке **ОК**. Таким образом, вы ввели условие, при котором в поле **Должность** могут вводиться только указанные значения.

11 В строке **Сообщение об ошибке** введите предложение «Такой должности нет, правильно введите данные».

12 В строке Значение по умолчанию введите слово **Доцент**.

13 Введите ограничения на данные в поле **Код преподавателя**. Здесь ограничения надо вводить не совсем обычным способом. Дело в том, что коды преподавателей не должны повторяться, а также должна быть обеспечена возможность их изменения (из-за последнего условия в этом поле нельзя использовать тип данных **Счетчик**, в котором данные не повторяются). Для выполнения второго условия измените тип поля **Код преподавателя** со **Счетчика** на **Числовой**, а для выполнения первого условия сделайте следующее:

– щелкните по строке параметра **Индексированное поле**.

Примечание – **Индекс** – это средство Access, ускоряющее поиск и сортировку данных в таблице. Ключевое поле (поле первичного ключа) таблицы индексируется автоматически. Не допускается создание индексов для полей типа MEMO и Гиперссылка или полей объектов OLE. Свойство **Индексированное поле** определяет индекс, создаваемый по одному полю. Индексированное поле может содержать как уникальные, так и повторяющиеся значения. Допускается создание произвольного количества индексов;

– выберите в списке пункт **Да** (совпадения не допускаются).

14 Для удобного заполнения таблицы данными воспользуемся возможностями Мастера подстановок:

– измените тип поля **Дисциплина** с текстового на **Мастер подстановок**;

– отметьте пункт **Будет введен фиксированный набор значений**;

– щелкните по кнопке **Далее**;

– число столбцов – 1, в столбец 1 введите значения: Информатика, Экономика, Математика, Физика:

– щелкните по кнопке **Далее**; подпись оставьте прежней – **Дисциплина**;

– щелкните по кнопке **Готово**;

– перейдите в режим **Таблица**, щелкнув по кнопке на линейке во вкладке **Главная** в группе команд **Режим**, выбрав команду **Режим таблицы**. На вопрос о сохранении таблицы щелкните по кнопке **Да**.

Заполнение базы данных

Заполните таблицу данными в соответствии с таблицей 1.2 и проверьте реакцию системы на ввод неправильных данных в поле **Должность**.

Измените ширину каждого поля таблицы в соответствии с шириной данных.

Таблица 1.2

Код	Фамилия	Имя	Отчество	Дата рождения	Должность	Дисциплина
1	Истомин	Роман	Петрович	23.10.54	Доцент	Информатика
2	Миронов	Павел	Юрьевич	25.07.47	Профессор	Экономика
3	Гришин	Евгений	Сергеевич	05.12.67	Доцент	Математика
4	Сергеева	Ольга	Ивановна	12.02.72	Ассистент	Математика
5	Петрова	Татьяна	Ивановна	16.02.51	Доцент	Экономика
6	Игнатова	Татьяна	Павловна	30.05.66	Доцент	Информатика
7	Миронов	Алексей	Павлович	30.07.48	Доцент	Физика

Для изменения ширины полей таблицы в соответствии с шириной данных:

– щелкните в любой строке поля **Код преподав**ателя;

– выполните команду на вкладке ленты **Главная, Запись, Дополнительно, Ширина столбца;**

– в появившемся окне щелкните по кнопке **По ширине данных**;

– проделайте эту операцию с остальными полями.

Ввод, просмотр и редактирование данных посредством формы

Формы являются объектом БД, позволяющим пользователям вводить и изменять данные в таблицы без непосредственного доступа к самим таблицам. Формы могут быть созданы не только для одной таблицы, но для нескольких связанных между собой таблиц и разработанных запросов.

Выполните следующее.

1 С помощью Мастера форм создайте форму **Состав преподавателей**:

– откройте вкладку **Создание** на линейке;

– выполните команду **Формы, Другие формы**;

– в появившемся списке выберите пункт **Мастер форм**;

– щелкните по значку списка в нижней части окна;

– выберите из появившегося списка таблицу **Преподаватели**, **ОК**;

– в появившемся окне выберите поля, которые будут присутствовать в форме. В данном примере нужны все поля, поэтому щелкните по кнопке >>;

– щелкните по кнопке **Далее**;

– в появившемся окне уже выбран вид Форма в один столбец, **Далее**;

– в появившемся окне выберите стиль оформления. Для этого щелкните по слову, обозначающему стиль, либо перемещайте выделение стрелками вверх или вниз на клавиатуре. После выбора стиля щелкните по кнопке **Далее**;

– в появившемся окне задайте имя формы, набрав на клавиатуре **Со-**

став преподавателей. Остальные параметры в окне оставьте без изменений;

– щелкните по кнопке **Готово**.

Перед Вами откроется форма в один столбец. Столбец слева — это названия полей, столбец справа — данные первой записи (в нижней части окна в строке параметра Запись стоит цифра «1»). Для перемещения по записям надо щелкнуть в указателе перемещения по записям Записи с большими номерами или с меньшими.

2 Произведите фильтрацию данных по полю **Должность**:

– щелкните по записи **Доцент** поля **Должность**;

– выполните команду **Сортировка** и **фильтр**;

– щелкните по записи Информатика поля «Дисциплина»;

– выполните команду **Сортировка** и фильтр, **Фильтр по выделенному**; в форме останутся только записи о преподавателях – доцентах кафедры информатики.

3 Измените название поля **Дисциплина** на **Преподаваемая дисциплина**. Для этого:

– перейдите в режим конструктора, щелкнув по кнопке на вкладке **Главная** в группе команд **Режим**;

– щелкните правой кнопкой мыши в поле **Дисциплина** (на названии поля – оно слева, а строка справа с именем **Дисциплина** —это ячейка для данных, свойства которых не будем менять). В появившемся меню выберите пункт **Свойства**. На экране откроется окно свойств поля **Дисциплина**. Щелкните по строке с именем **Подпись**, т. е. там, где находится слово **Дисциплина**;

– сотрите слово **Дисциплина** и введите **Преподаваемая дисциплина**;

– для просмотра результата перейдите в режим формы, выполнив команду перехода в **Режим формы**.

1.2 Лабораторная работа № 2. Формирование запросов для однотабличной базы данных

Цель: освоить создание запросов к БД, с помощью которых можно задавать вопросы о БД и получать на них ответы.

Формирование и использование запросов на выборку

1 На основе таблицы **Преподаватели с**оздайте простой запрос на выборку, в котором должны отображаться фамилии, имена, отчества преподавателей и их должности.

2 Данные запроса отсортируйте по должностям.

3 Сохраните запрос.

4 Создайте запрос на выборку с параметром, в котором должны отображаться фамилии, имена, отчества преподавателей и преподаваемые ими дисциплины, а в качестве параметра задайте фамилию преподавателя и выполните этот запрос для преподавателя *Гришина.*

Выполните следующее.

1 Для создания простого запроса:

– откройте вкладку **Создание** и выберите из группы команд **Другие**, команду **Мастер запросов**;

– в открывшемся окне из появившихся пунктов окна **Новый запрос** выберите **Простой запрос** и щелкните по кнопке **ОК**;

– в появившемся окне в строке **Таблицы/запросы** выберите таблицу **Преподаватели** (если других таблиц или запросов не было создано, она будет одна в открывающемся списке); в окне **Доступные поля** переведите выделение на параметр **Фамилия**;

– щелкните по кнопке **>**. Слово **Фамилия** перейдет в окно **Выбранные поля**;

– аналогично в окно **Выбранные поля** переведите поля **Имя**, **Отчество**, **Должность** (порядок важен – в таком порядке данные и будут выводиться);

– щелкните по кнопке **Далее>**;

– в строке параметра **Задайте имя запроса** введите новое имя **Должности преподавателей**;

– щелкните по кнопке **Готово**. На экране появится таблица с результатами запроса.

2 Для сортировки данных:

– щелкните в любой строке поля **Должность**;

– отсортируйте данные по убыванию. Для этого выполните команду **Записи, Сортировка, Сортировка по убыванию**.

3 Для сохранения запроса:

– выполните команду **Файл**, **Сохранить**;

– закройте окно запроса.

4 Для создания запроса на выборку с параметром:

– создайте запрос на выборку для следующих полей таблицы **Преподаватели**: **Фамилия**, **Имя**, **Отчество**, **Преподаваемая дисциплина** аналогично тому, как это делалось в п.1; задайте имя запросу **Преподаваемые дисциплины**;

– щелкните по кнопке **Готово**. На экране появится таблица с результатами запроса;

– перейдите в режиме конструктора, щелкнув по кнопке **Конструктор** на вкладке **Главная** в группе команд **Режим**;

– в строке параметра **Условия отбора** для поля **Фамилия** введите фразу (скобки тоже вводить): [***Введите фамилию преподавателя***];

– выполните запрос, щелкнув по кнопке **Выполнить** на вкладке **Конструктор** в группе команд **Результаты**.

Примечание – Вышеописанным способом запрос выполняется только в режиме конструктора. Для того чтобы выполнить запрос из другого режима:

– откройте вкладку *Запросы*;

– выделите требуемый запрос и щелкните по кнопке <Открыть>;

– в появившемся окне введите фамилию *Гришин* и щелкните по кнопке <ОК>. На

экране появится таблица с данными о преподавателе *Гришине* – его имя, отчество и преподаваемая им дисциплина;

– сохраните запрос; закройте окно запроса.

1.3 Лабораторная работа № 3. Разработка модели и создание структуры реляционной базы данных

Цель: преобразовать созданную однотабличную БД в многотабличную, состоящую из набора связанных между собой таблиц.

Для создания реляционной базы данных выполните следующее.

1 Создайте структуры таблиц *Студенты*, *Дисциплины*, *Оценки*:

– на вкладке *Создание* в группе команд *Таблицы* щелкните по кнопке <Таблицы>, при этом в линейке добавится вкладка ***Режим таблицы***;

– в окне *Сохранение* введите имя таблицы: *Студенты*. Выберите режим *Конструктор*. В результате проделанных операций открывается окно таблицы в режиме конструктора, в котором следует определить поля таблицы;

– определите поля таблицы в соответствии с данными таблицы 1.3.

Таблица 1.3

Имя поля	Тип данных	Размер поля
Код студента	Числовой	Целое
Фамилия	Текстовый	15
Имя	Текстовый	12
Отчество	Текстовый	15
Номер группы	Числовой	Целое
Телефон	Текстовый	9
Стипендия	Логический	Да/Нет

В качестве ключевого поля задайте ***«Код студента»***. Закройте таблицу с сохранением.

2 Аналогично создайте структуру таблицы ***Дисциплины*** в соответствии с данными таблицы 1.4.

Таблица 1.4

Имя поля	Тип данных	Размер поля
Код дисциплины	Числовой	Целое
Название дисциплины	Текстовый	30

В качестве ключевого поля задайте **Код дисциплины**.

3 Создайте структуру таблицы ***Оценки*** в соответствии с данными таблицы 1.5.

Таблица 1.5

Имя поля	Тип данных	Размер поля
Код студента	Числовой	Целое
Код дисциплины	Числовой	Целое
Оценки	Числовой	Байт

В этой таблице задавать ключевое поле не надо, так как данные во всех полях могут повторяться.

4 Измените структуру таблицы ***Преподаватели***. В качестве ключевого поля задайте **Код дисциплины**.

Структура таблицы ***Преподаватели*** создана в лабораторной работе № 1 и заполнена данными, поэтому для работы используйте эту таблицу с одним лишь изменением: в структуру таблицы надо добавить поле **Код дисциплины** (в режиме конструктора). Тип данных размер поля установить в соответствии с данными таблицы 1.4.

5 Разработайте схему данных, т. е. создайте связи между таблицами. Для этого:

– щелкните по кнопке **Схема данных** на вкладке **Работа с базами данных**. На экране появится окно **Схема данных**;

– используя контекстное меню, добавьте таблицы: щелкните по кнопке **Добавить таблицу**;

– в появившемся окне будет выделено название одной таблицы. Щелкните по кнопке **Добавить**;

– переведите выделение на имя следующей таблицы и щелкните по кнопке **Добавить**. Аналогично добавьте оставшиеся две таблицы;

– закройте окно, щелкнув по кнопке **Закрыть**;

– создайте связь между таблицами **Дисциплины** и **Оценки**. Для этого подведите курсор мыши к полю **Код дисциплины** в таблице **Дисциплины**, щелкните левой кнопкой мыши и, не отпуская ее, перетащите курсор на поле **Код дисциплины** в таблицу **Оценки**, а затем отпустите кнопку мыши. На экране откроется окно **Связи**;

– установите флажок («галочку») в свойстве **Обеспечение целостности данных**, щелкнув по нему;

– установите флажок в свойствах **Каскадное обновление связанных полей** и **Каскадное удаление связанных записей**.

Примечание – Задание каскадного обновления связанных полей и каскадного удаления связанных записей позволит вам отредактировать записи только в таблице *Дисциплины,* а в таблице *Оценки* эти действия будут со связанными записями выполняться автоматически. Например, если вы удалите из таблицы *Дисциплины* один предмет, то в таблице *Оценки* удалятся все строки, связанные с этим предметом;

– щелкните по кнопке **Создать**. Связь будет создана;

– аналогично создайте связи между полем **Код дисциплины** в таблице **Дисциплины** и полем **Код дисциплины** в таблице **Преподаватели**, а также между полем **Код студента** в таблице **Студенты** и полем **Код студента** в таблице **Оценки**. Результат представлен на рисунке 1.1;

– закройте окно схемы данных, ответив **ДА** на вопрос о сохранении макета.

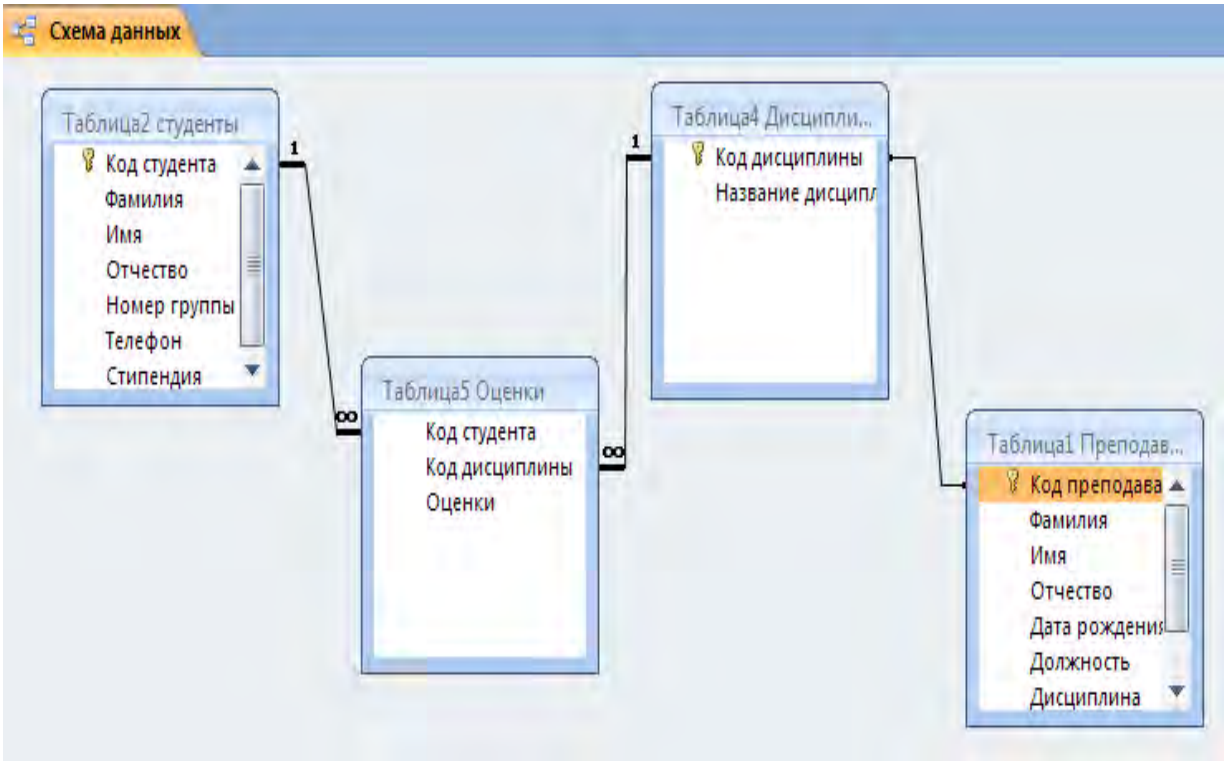


Рисунок 1.1

Создание и использование форм для ввода данных в таблицы

Выполните следующее.

1 Создайте формы **Студенты**, **Дисциплины**, **Оценки**. Заполните данными соответствующие таблицы, используя формы.

2 Для создания формы **Студенты**:

– откройте вкладку **Создание**;

– щелкните по кнопке **Другие формы** и далее выберите **Мастер форм**;

– в открывающемся списке выберите таблицу **Студенты**;

– щелкните по кнопке **вид**;

– выберите вид формы: **ленточная**;

– щелкните по кнопке **Готово**. Форма для ввода данных создана.

Заполните данными, приведенными в таблице 1.6, таблицу **Студенты** посредством формы (данные таблицы можно изменить по своему усмотрению).

Таблица 1.6

Код студента	Фамилия	Имя	Отчество	Номер группы	Стипендия
1	2	3	4	5	6
1	Арбузов	Николай	Николаевич	151	Да
2	Киршин	Петр	Валерьевич	151	Да
3	Кривцов	Сергей	Николаевич	151	Нет

Окончание таблицы 1.6

1	2	3	4	5	6
4	Крылова	Елена	Петровна	151	Да
5	Кульчий	Григорий	Викторович	151	Да
6	Патрикеев	Олег	Борисович	152	Нет
7	Перлов	Кирилл	Николаевич	152	Нет
8	Соколова	Наталия	Петровна	152	Нет
9	Степанская	Ольга	Витальевна	152	Да
10	Тимофеев	Сергей	Иванович	152	Да

Закройте форму, задав ей имя **Студенты**.

3 Аналогично создайте формы **Дисциплины** и **Оценки**.

Заполните данными, приведенными в таблице 1.7, таблицу базы данных **Дисциплины** посредством формы и закройте форму, задав ей имя **Дисциплины**.

Таблица 1.7

Код дисциплины	Название дисциплины
1	Информатика
2	Математика
3	Физика
4	Экономика

Заполните данными таблицу **Оценки**, применяя форму. Закройте форму, задав ей имя **Оценки**.

В таблицу **Оценки** внесите 15…20 записей. Данные по вашему выбору для конкретного студента могут содержать оценки по разным дисциплинам.

Закройте файл, сохранив все изменения.

1.4 Лабораторная работа № 4. Формирование сложных запросов

Цель: научиться создавать запросы для извлечения данных из таблиц, производить их сортировку, фильтрацию и преобразование по заданному алгоритму.

Теоретические сведения

Запросы используются для просмотра, изменения и анализа данных различными способами. Также они могут применяться в качестве источников записей для форм, отчетов. В Microsoft Access есть несколько типов запросов.

Запросы на выборку. Запрос на выборку является наиболее часто используемым типом запроса. Запросы этого типа возвращают данные из одной или нескольких таблиц и отображают их в виде таблицы, записи в которой можно обновлять (с некоторыми ограничениями). Запросы на выборку можно также использовать для группировки записей и вычисления сумм, средних значений, подсчета записей и нахождения других типов итоговых значений.

Запросы с параметрами. Запрос с параметрами – это запрос, при выполнении отображающий в собственном диалоговом окне приглашение ввести данные.

Перекрестные запросы. Перекрестные запросы используют для расчетов и представления данных в структуре, облегчающей их анализ. Перекрестный запрос подсчитывает сумму, среднее, число значений или выполняет другие статистические расчеты, после чего результаты группируются в виде таблицы по двум наборам данных, один из которых определяет заголовки столбцов, а другой – заголовки строк.

Запросы на изменение. Запросом на изменение называют запрос, который за одну операцию изменяет или перемещает несколько записей.

Запросы SQL. Запрос SQL – это запрос, создаваемый при помощи инструкций SQL. Язык SQL (Structured Query Language) используется при создании запросов, а также для обновления и управления реляционными базами данных, такими как базы данных Microsoft Access.

Когда пользователь создает запрос в режиме конструктора запроса, Microsoft Access автоматически создает эквивалентную инструкцию SQL. Пользователь имеет возможность просматривать и редактировать инструкции SQL в режиме SQL. После внесения изменений в запрос в режиме SQL его вид в режиме конструктора может измениться.

Окно конструктора запроса (рисунок 1.2). Окно конструктора запроса разделено на две части. Верхняя область окна – это панель таблиц, каждая из которых представлена списком полей, нижняя панель окна – это бланк или таблица запроса, его называют QBE-областью (Query by Example – запрос по образцу). В нижней панели указываются параметры запроса : имена полей, участвующих в запросе, а также условия отбора и сортировки записей. Для перемещения между панелями можно использовать клавишу <**F6**>.

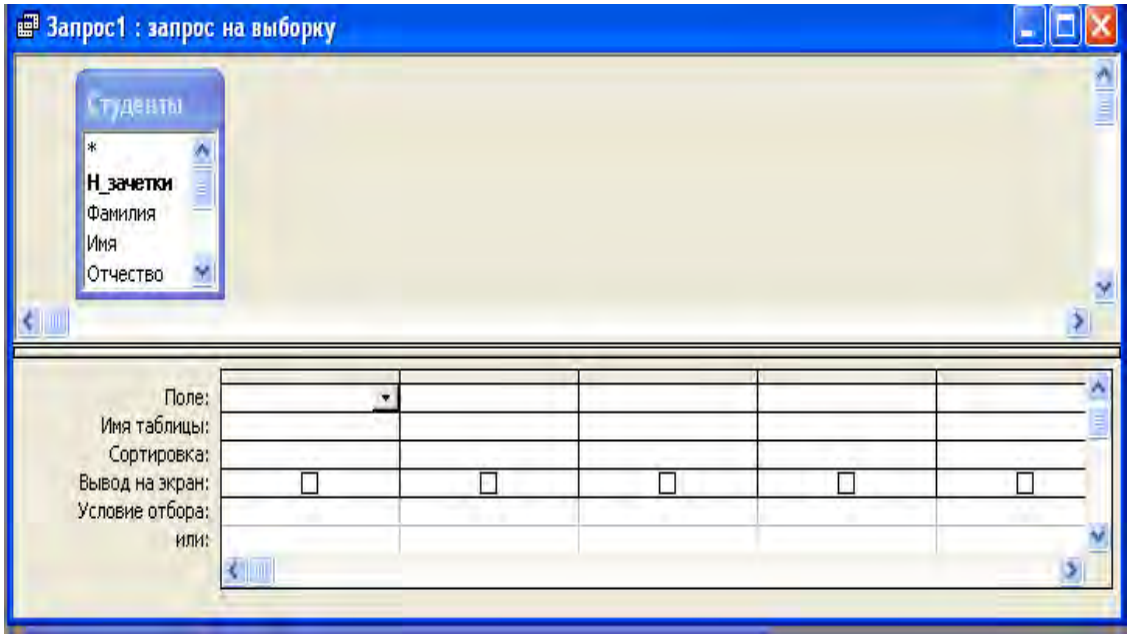


Рисунок 1.2 – Окно конструктора запроса

Практические задания

Используется база данных Деканат.

1 Разработайте запрос с параметрами о студентах заданной группы, в котором при вводе в окно параметров номера группы (в примере это 151 или 152) на экран должен выводиться состав этой группы.

2 Создайте запрос, в котором выводятся оценки студентов заданной группы по заданной дисциплине.

3 Создайте перекрестный запрос, в результате которого создастся выборка, отражающая средний балл по дисциплинам в группах.

4 Создайте запрос на удаление отчисленных студентов.

5 Разработайте запрос на создание базы данных отличников.

Для всех созданных вами запросов разработайте формы.

Выполните следующее.

1 Создание запроса с параметрами о студентах заданной группы:

– откройте вкладку **Создание** на ленте*;*

– щелкните по кнопке **Мастер запросов**;

– выберите **Простой запрос** и щелкните по кнопке ОК;

– в появившемся окне в строке **Таблицы/запросы** выберите из списка таблицу **Студенты;**

– перенесите все поля из окна **Доступные поля** в окно **Выбранные поля**, щелкнув по кнопке >>;

– щелкните по кнопке **Далее**. Выводить надо все поля, поэтому еще раз щелкните по кнопке **Далее**;

– выберите подробный или итоговый отчет: подробный; щелкните по кнопке **Далее**;

– в появившемся окне задайте имя запроса Группа*;*

– щелкните по кнопке **Готово**. На экране появится таблица с данными запроса. Но вам надо, чтобы при выполнении запроса выяснялся номер группы. Для этого перейдите в режим конструктора;

– в строке **Условия отбора** для поля **Номер группы** введите фразу: **[*Введите номер группы*];**

– выполните запрос, щелкнув по кнопке **Выполнить** на ленте**;**

– в появившемся окне введите *151* и щелкните по кнопке **ОК**. На экране появится таблица с данными о студентах 151-й группы;

– сохраните запрос и закройте таблицу запроса.

2 Создание запроса, в котором выводятся оценки студентов заданной группы по заданной дисциплине:

– на вкладке **Создание** щелкните по кнопке **Мастер запросов**;

– выберите **Простой запрос** и щелкните по кнопке **ОК**;

– выберите таблицу **Студенты** и перенесите поля **Фамилия**, **Имя**, **Отчество**, **Номер группы** в окно **Выделенные поля**. В таблице **Дисциплины** выберите поле **Название дисциплины**;

– в таблице **Оценки** выберите поле **Оценки**.

Вы сформировали шесть полей запроса — они связаны между собой посредством схемы данных;

– щелкните по кнопке **Далее**>, затем в появившемся окне снова щелкните по кнопке **Далее**;

– в появившемся окне задайте имя запроса **Оценки группы**, затем щелкните по ячейке **Изменение макета запроса** – это позволит сразу перейти в режим конструктора;

– щелкните по кнопке **Готово**;

– в строке **Условия отбора** для поля **Номер группы** введите фразу: **[*Введите номер группы*]**;

– в строке **Условия отбора** для поля **Название дисциплины** введите фразу: [**Введите название дисциплины]**;

– выполните запрос;

– в первом появившемся окне введите *152*, затем щелкните по кнопке **ОК**, во втором – введите ***Информатика*** и щелкните по кнопке **ОК**. На экране появится таблица со списком 152-й группы и оценками по информатике;

– сохраните запрос и закройте таблицу запроса.

3 Создание перекрестного запроса о среднем балле в группах по дисциплинам. Такой запрос строится на основе одной таблицы или одного запроса, в связи с чем надо сначала сформировать запрос, в котором были бы поля **Номер группы, Название дисциплины и Оценки**. Для этого:

– на вкладке **Создание** щелкните по кнопке **Мастер запросов**;

– выберите **Простой запрос** и щелкните по кнопке **ОК**;

– выберите из таблицы **Студенты** поле **Номер группы**;

– выберите из таблицы **Дисциплины** поле Название дисциплины ;

– выберите из таблицы **Оценки** поле **Оценки**;

– щелкните по кнопке **Далее**, затем в появившемся окне снова щелкните по кнопке **Далее**;

– в появившемся окне задайте имя запроса **Дисциплины и оценки группы***;*

– щелкните по кнопке **Готово**;

– сохраните запрос и закройте таблицу запроса.

Теперь можно создавать перекрестный запрос. Для этого:

– на вкладке ***Запросы*** щелкните по кнопке **Мастер запросов**;

– выберите **Перекрестный запрос** и щелкните по кнопке **ОК**;

– щелкните по ячейке **Запросы**, выберите ***Дисциплины и оценки группы*** и щелкните по кнопке Далее;

– выберите поле **Название дисциплины** и с помощью кнопки переместите в окно **Доступные поля**;

– выберите поле **Номер группы** и щелкните по кнопке **Далее**;

– выберите функцию среднее и щелкните по кнопке **Далее**;

– задайте название запроса **Средние оценки** и щелкните по кнопке **Готово**. Откроется таблица перекрестного запроса. Обратите внимание на то, что Access создает еще итоговое значение средних оценок по дисциплинам;

– закройте таблицу запроса.

4 Для создания запроса на отчисление студента гр. 152 *Перлова Кирилла Николаевича:*

– на вкладке **Создание** щелкните по кнопке **Мастер запросов**;

– выберите **Простой запрос**;

– в таблице **Студенты** выберите поля **Фамилия**, **Имя**, **Отчество**, **Номер группы**;

– щелкните по кнопке **Далее**, затем в появившемся окне выберите **Подробный отчет** и снова щелкните по кнопке **Далее**; в появившемся окне задайте имя запроса ***Отчисленные студенты***;

– щелкните по ячейке **Изменить макет запроса**;

– щелкните по кнопке **Готово**;

– в строке **Условия отбора** введите: в поле **Фамилия**» – *Перлов,* в поле **Имя** – *Кирилл,* в поле **Отчество** – *Николаевич,* в поле **Номер группы** – *152*;

– на вкладке ленты **Конструктор** в группе команд **Тип запроса** выберите команду **Удаление**;

– просмотрите удаляемую запись, выбрав вкладку **Главная**, выполнив команду перехода в **Режим таблицы**;

– если отчисляемый студент выбран правильно, то перейдите в режим конструктора и выполните запрос. Если условия отбора сделаны неправильно, измените их; закройте запрос;

– откройте форму **Студенты** и удостоверьтесь в удалении записи о студенте Петрове;

– закройте форму.

5 Для создания запроса на создание базы данных отличников:

– на вкладке **Создание** щелкните по кнопке **Мастер запросов**. Выберите **Простой запрос**;

– в таблице **Студенты** выберите поля Фамилия, Имя, Отчество и Номер группы, а в таблице ***Оценки*** – поле **Оценки**; щелкните по кнопке **Далее**, затем в появившемся окне выберите **Подробный отчет** и снова щелкните по кнопке **Далее**;

– в появившемся окне задайте имя запроса **Отличники**;

– щелкните по кнопке **Готово**.

Будем считать отличниками тех студентов, которые набрали за четыре экзамена 20 баллов. Операция группировки позволит просуммировать оценки студентов по всем экзаменационным дисциплинам;

– для выполнения групповых операций щелкните на вкладке ленты в группе команд **Показать или скрыть** по кнопке **Итоги** в **Режиме конструктора**;

– в строке **Групповые операции** поля **Оценки** щелкните по ячейке **Групповые операции**.

– откройте раскрывающийся список и выберите функцию **SUM**;

– в строке **Условия отбора поля Оценки** введите *20;*

– просмотрите создаваемую базу, выполнив команду перехода в **Режим таблицы**; перейдите в режим конструктора; выполните команду **Создание таблицы** из группы команд **Тип запроса**;

– задайте имя таблицы ***Студенты-отличники*** и щелкните по кнопке **ОК**; подтвердите создание таблицы;

– закройте с сохранением запрос;

– откройте вкладку **Запросы** в области переходов, выберите созданный запрос **Отличники***;*

– подтвердите создание таблицы **Студенты-отличники**. Удостоверьтесь в правильности создания таблицы;

– закройте таблицу.

1.5 Лабораторная работа № 5. Создание сложных форм и отчетов

Цель: научиться создавать формы и отчеты, используя поля из нескольких связанных таблиц.

Задание

1 Разработайте сложную форму, в которой с названиями дисциплин были бы связаны подчиненная форма **Студенты** и подчиненная форма **Оценки студентов**.

2 Вставьте в форму диаграмму, графически отражающую оценки студентов.

3 Отредактируйте вид осей диаграммы.

Выполните следующее.

1 Для создания сложной формы:

– на вкладке **Формы** щелкните по кнопке **Другие формы**, выберите **Мастер форм**;

– в таблице **Дисциплины** выберите поле **Название дисциплины**;

– в таблице **Студенты** выберите поля **Код студента**, **Фамилия, Имя, Отчество, Номер группы**;

– в таблице **Оценки** выберите поле **Оценки** и щелкните по кнопке **Далее**;

– в появившемся окне выберите вид представления данных;

– оставьте табличный вариант подчиненной формы и щелкните по кнопке **Далее**;

– выберите нужный стиль оформления и щелкните по кнопке **Далее**;

– задайте название формы **Дисциплины и оценки**;

– щелкните по кнопке **Готово** и просмотрите полученную форму.

2 Если не удовлетворяет расположение полей на экране. Измените их положение, оставив место для диаграммы. Для этого:

– перейдите в режим конструктора;

– стандартными средствами Windows (технология *drag-and-drop*) измените размеры подчиненной формы так, чтобы были видны все данные. Для этого надо (как правило, многократно) переключаться из режима конструктора в режим формы, смотреть на полученный результат и, если он не подходит, снова корректировать в режиме конструктора. Ширину столбцов в подчинен-

ной форме можно изменить только в режиме формы.

3 Для того чтобы вставить в форму диаграмму оценок студентов по заданным дисциплинам, необходимо:

– переключиться в режим конструктора;

– выполнить команду **Элементы управления, Вставить диаграмму**;

– на свободном месте формы растянуть прямоугольник для диаграммы;

– выбрать таблицу **Оценки** и щелкнуть по кнопке **Далее**;

– выбрать поля **Код студента** и **Оценки**;

– щелкнуть по кнопке **Далее**;

– выбрать тип диаграммы **Гистограмма** (предлагаемый по умолчанию) и щелкнуть по кнопке **Далее**;

– выбрать тип отображения данных, щелкнуть по кнопке **Далее**;

– в строке ***Поля формы*** и в строке ***Поля диаграммы*** оставить все по умолчанию;

– задать название диаграммы ***Диаграмма оценок*** (так как уже задали надпись для диаграммы) и щелкнуть по кнопке **Далее**.

4 Отредактируйте вид осей диаграммы. Для этого:

– дважды щелкните по диаграмме;

– дважды щелкните по значениям вертикальной оси;

– выберите вкладку **Шкала**;

– уберите «галочку» у надписи **Минимальное значение**, а в ячейке справа от этого названия введите 1;

– уберите «галочку» у надписи **Максимальное значение**, а в ячейке справа от этого названия введите 5;

– уберите «галочку» у надписи **Цена основных делений**, а в ячейке справа от этого названия введите 1 и щелкните по кнопке **ОК**;

– расширьте область диаграммы, перетащив правую границу окна диаграммы несколько правее (подведя курсор к правой границе до появления двойной стрелки и нажав левую кнопку мыши);

– закройте окно **Microsoft Graph**, переведя курсор мыши на любое место формы и щелкнув левой кнопкой мыши;

– перейдите в режим формы и просмотрите форму для разных дисциплин (щелкая по кнопке перехода к следующей записи в нижней части формы). Вы увидите изменение названий дисциплин, а также оценок студентов по данным дисциплинам и изменение диаграмм, отображающих эти оценки;

– закройте форму.

Технология создания отчетов идентична способам работы по созданию форм. Но они имеют иное функциональное назначение – служат для форматированного вывода данных на печатающие устройства.

2 Основные приемы работы с пакетом MATLAB

2.1 Лабораторная работа № 6. Графический интерфейс пользователя. Простейшие вычисления

Цель: изучить основные положения по рассматриваемой тематике, используя приведенные в работе сведения и справочную систему MATLAB, выполнить практические задания.

Командное окно

Задание 1

Выполните следующее.

1 Запустите MATLAB.

2 Используя команду меню **View**, установите режим отображения только командного окна.

3 Используя средства **ОС**, опробуйте способы изменения размеров окна.

4 Установите текущей рабочую папку.

5 Введите в командном окне:

x=2 и далее ↵ **(**символ нажатия клавиши ввод)

y=3; ⇒ ↵

z=(x+y +5)/2-x^2 ↵

Проанализируйте результаты, отображаемые в командном окне после ввода. Применение точки с запятой позволяет подавить вывод результатов после нажатия клавиши ↵**.**

После многократного повторения вычислений, когда строки сдвигаются вверх, так что верхняя строка покидает область видимости, в самом низу окна появляется свободная строка для ввода новых данных. Эта строка содержит знак приглашения **>>.**

Информация, покинувшая видимую часть окна, не исчезает. Ее можно увидеть снова, если осуществить вертикальную протяжку содержимого командного окна, используя полосу протяжки, или с помощью следующих клавиш клавиатуры: PageUp, PageDown, Ctrl+Home и Ctrl+End.

Клавиши **Стрелка вверх** и **Стрелка вниз** в любом текстовом редакторе осуществляют перемещение курсора вверх-вниз и вертикальную протяжку содержимого окна, *в MATLAB работают иначе.* Эти клавиши позволяют вернуть в строку ввода ранее введенную в командное окно входную информацию, как правило, данная возможность используется для редактирования.

В итоге можно сказать, что видимая информация в окне системы MATLAB располагается в двух разных зонах: зоне просмотра и зоне редактирования (рисунок 2.1).

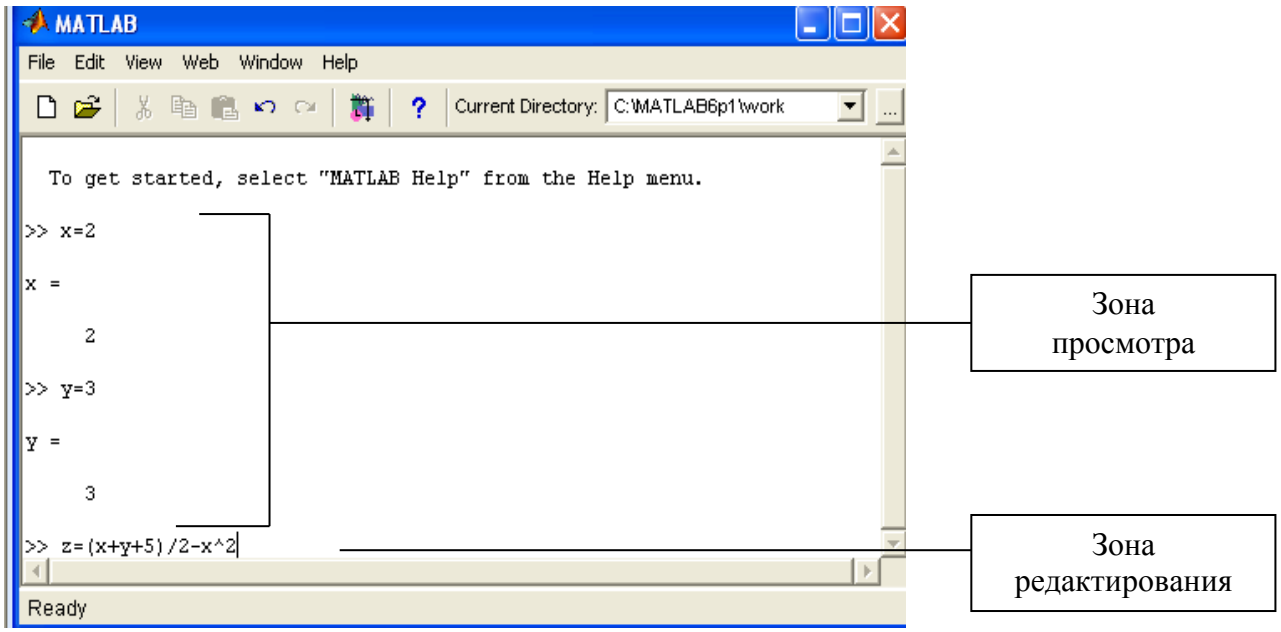


Рисунок 2.1

В зоне просмотра ничего нельзя исправить, хотя в нее можно поместить курсор, однако реакцией на ввод с клавиатуры будет перемещение курсора в строку ввода, расположенную в зоне редактирования. В зоне просмотра можно выделять с помощью мыши любую информацию и помещать ее в буфер обмена, чтобы потом вставить в документ текстового редактора либо в строку ввода.

Задание 2

Выполните следующее.

1 С использованием ↑↓ поместите в строку ввода ранее введенное выражение **y=3;** Редактируйте **y=3*x;** Нажмите↵**.**

2 В зоне просмотра выделите **z=(x+y +5)/2-x^2**, поместите в буфер обмена. Вставьте в строку ввода. Измените **z=sqrt((x+y +5)/2)-x^2**.

Зона редактирования обычно занимает одну (последнюю) строку командного окна, в которой находится знак приглашения >>. Она называется строкой ввода. При необходимости эту «логическую строку ввода» можно распространить на несколько физических строк командного окна. Для ввода с показом вводимой информации на следующих физических строках используется помещение в место разрыва строки трех или более точек. После нажатия клавиши **Enter** вводимая информация отображается на экране на следующей физической строке. Однако в этом случае зона редактирования распространяется только на самую последнюю строку (теперь она уже не содержит знака приглашения).

Задание 3

Выполните следующее.

1 В командном окне введите

>>r=x+5+6+…..↵ y+2+3

2 Самостоятельно попытайтесь найти вариант редактирования введенного в нескольких строках выражения. Преобразуйте последнее выражение

к следующему виду:

```
>> r=x+5+6+11+...
6/y+2+3+10
```

Рабочее пространство

Все значения переменных, вычисленные в течение текущего сеанса работы, сохраняются в специально выделяемой области памяти компьютера, называемой рабочим пространством системы MATLAB (Matlab WorkSpace).

После завершения сеанса работы с MATLAB все ранее вычисленные переменные теряются. Чтобы сохранить содержимое рабочего пространства, нужно выполнить команду **File⇒Save Workspace As…**, после чего в стандартном диалоговом окне указываются имя папки и имя файла. Расширение имени файла должно быть **mat,** поэтому такие файлы принято называть MAT-файлами.

Вместо рассмотренной команды меню можно непосредственно в командном окне системы MATLAB набрать команду

save путь_к_файлу\имя_MAT_файла

и нажать клавишу **Enter.** Если не указывать пути, то файл будет сохранен в текущей рабочей папке, которая показана на полосе инструментов окна системы MATLAB.

Для загрузки в память компьютера ранее сохраненного рабочего пространства нужно выполнить команду

File⇒Open…

Вместо рассмотренной, можно набрать команду

load имя_MAT_файла

Можно также из записанного на диске **MAT_файла** считывать в рабочее пространство значения отдельных переменных

load имя_MAT_файла имя_1, имя_2,…

Если MAT-файл указан без полного пути к нему, то он должен находиться в текущей папке.

Командой **clc** можно стереть видимое содержимое командного окна, однако это не затронет содержимого рабочего пространства.

По мере увеличения рабочего пространства эффективность работы снижается. Для того чтобы удалить переменные из памяти компьютера, используются команды:

clear имя_1, имя_2… – удаляются указанные переменные;

clear – удаляются все переменные.

Для проверки списка переменных, которые находятся в рабочем пространстве, используются команда **who**.

Для просмотра значения любой переменной из текущего рабочего пространства наберите ее имя и нажмите **Enter**.

Задание 4

Выполните следующее.

1 Сохраните ранее созданные переменные в рабочей папке в файле с име-

нем Test_1.mat (имя файла можете выбрать по своему усмотрению). Используйте для сохранения команду **save…**.

2 Удалите из рабочей области одну или две переменных. Сохраните после удаления рабочую область в файле Test_2.mat.

3 Удалите из рабочей области все переменные. Очистите командное окно.

Вещественные числа, типы данных

Основным типом данных, с которыми производятся вычисления в среде MATLAB, являются конечные десятичные дроби, приближающие с заданной точностью произвольные вещественные числа.

Вещественные числа задаются мантиссой и показателем степени.

Задание 5

Выполните следующее.

В командном окне введите вещественные числа

r1=2.851034e+8; r2=-456.32145; r3=- 45678,234; r4=185e-1; r5=4.5; r6=-123.

Этот основной тип данных называется **double,** задается по умолчанию. Для хранения отводится 8 байт памяти.

Существует возможность понизить точность представления вещественных чисел, явно перейдя к типу данных **single. x=single(3.5)**.

Задание 6

Выполните следующее.

1 Установите значения минимального и максимального по модулю вещественных чисел, представимых в системе MATLAB, присваивая переменным **r7** и **r8** значения констант **realmax** и **realmin.**

2 Переведите переменные **r7** и **r8** в тип данных **single** и попытайтесь интерпретировать результаты.

3 Установите, как меняется отображение чисел в командном окне после изменения используемого по умолчанию формата **short** командой **format long**.

4 Установите назначение формата **format rat**.

Использование точки с запятой

Если в командном окне вводится выражение и нажимается клавиша **Enter**, производится вычисление и результат выводится в командном окне. Если в конце введенного выражения поставить точку с запятой, то результат вычисления не выводится на экран. Таким образом, точка с запятой ***подавляет вывод*** результатов вычисления на экран. Кроме того, если мы хотим за один раз, т. е. одним нажатием клавиши **Enter**, вычислить несколько выражений и присвоить значения нескольким переменным вычисляемым, то эти выражения надо отделять друг от друга точкой **с запятой**. Таким образом, точка с запятой работает как ***разделитель***.

Задание 7

Выполните следующее.

1 Введите выражение **d=round(r5+0.2)**, используя полученный результат, установите назначение функции **round**. Для этого самостоятельно проанализируйте результат или получите справочную информацию (по любой функции), выполнив команду **help имя_функции**.

2 Используя отведенную для этого клавиш, верните в строку ввода ранее введенную команду **d=round(r5+0.2).** Отредактируйте выражение, приведите его к виду **d1=floor(r5+0.2).**

3 После нажатия клавиши **Enter** для просмотра значения выражения введите **d1**↵ или применив функцию **disp (d1)** ↵. Назначение функции выясните, прибегнув к справочной системе.

4 Сохраните переменные, созданные при выполнении заданий, в файле ***Test_3.mat***.

Работа с системой MATLAB в режиме прямых вычислений

Работа с системой в режиме прямых вычислений носит диалоговый характер. Вычисляемое выражение вводится путем набора на клавиатуре и нажатия клавиши **ENTER**. При этом действует ростейший строчный редактор.

Важными командами системы являются:

HELP – помощь; **DEMO** – демонстрация; **INFO** – информация.

MATLAB содержит несколько системных переменных, в том числе: **pi** –число «пи»; **inf** – значение машинной бесконечности; **ans** – переменную, хранящую результат последней операции.

Если запись оператора не заканчивается символом «**;**», то результат выводится в командное окно, в противном случае – не выводится.

Если оператор не содержит знака присваивания «=», то значение результата присваивается системной переменной **ans**.

При работе с числовыми данными можно задавать различные форматы представления чисел (при этом все вычисления проводятся с предельной, так называемой двойной точностью).

Для установки формата представления чисел используется команда «**format name**», где «**name**» – имя формата.

Система MATLAB позволяет вычислять различные математические функции. Следующие элементарные алгебраические функции имеют в качестве аргумента одно или два действительных (x, y) или одно комплексное (z) число:

abs(z) – вычисление модуля комплексного числа z или абсолютного значения действительного числа z;

angle(z) – вычисление аргумента z;

sqrt(z) – вычисление квадратного корня;

real(z) – вычисление действительной части z;

imag(z) – вычисление мнимой части z;

round(z) – округление до целого;

fix(z) – округление до ближайшего целого в сторону нуля;

floor(z) – округление до ближайшего целого в сторону отрицательной бесконечности;

sign(z) – вычисление функции знака;

rem(x, y) – вычисление остатка от деления x на y;

exp(z) – вычисление e в степени z;

log(z) – вычисление натурального логарифма числа z;

log10(z) – вычисление десятичного логарифма числа z.

Система MATLAB предоставляет возможности для вычисления следующих тригонометрических и обратных тригонометрических функций:

sin(z) – вычисление синуса;

cos(z) – вычисление косинуса;

tan(z) – вычисление тангенса;

asin(z) –вычисление арксинуса;

acos(z) – вычисление арккосинуса;

atan(z) – вычисление арктангенса;

atan2(y, x) – вычисление арктангенса по координатам точки.

Контрольные вопросы

1 Как настроить режим отображения окна Matlab?

2 В каких зонах располагается в окне введенная информация? Можно ли разбивать выражение при вводе на несколько строк?

3 Как отредактировать ранее введенную в командном окне информацию?

4 Где сохраняются переменные текущего сеанса работы?

5 Способы загрузки сохраненных переменных в память компьютера.

6 Типы числовых данных, используемых в Matlab. Форматы их отображения в командном окне. Удаление переменных из памяти компьютера.

7 Варианты использования точки с запятой.

8 Назначение команд **who** и **whos**.

2.2 Лабораторная работа № 7. Одномерные и двумерные числовые массивы

Цель: изучить способы создания одномерных и двумерных массивов. Вычислить функции от массивов.

Теоретические сведения

В среде MATLAB можно производить вычисления с набором вещественных (или комплексных) чисел так же легко, как и с одиночными числами. Это является одним из самых заметных и важных преимуществ системы MATLAB над другими программными пакетами, ориентированными на вычисления и программирование.

Именованные наборы чисел в различных языках программирования традиционно называют *массивами.* Всему массиву присваивается одно имя, а доступ

к отдельным *элементам массива* осуществляется по целочисленному *индексу,* т. е. по номеру элемента в массиве.

В зависимости от количества индексов, с помощью которых осуществляется доступ к отдельным элементам, массивы разделяются на *одномерные* (единственный индекс), *двумерные* (два индекса) и массивы больших размерностей (три индекса и более). Последние массивы принято называть *многомерными*.

Числовые массивы (вещественные или комплексные) являются массивами элементов типа **double**.

Одномерные числовые массивы – это линейные наборы чисел, в которых позиция каждого элемента задается единственным числом – его номером. Можно говорить о первом элементе массива, о втором и т. д.

Для создания одномерного массива можно использовать *операцию конкатенации.* Эта операция обозначается с помощью квадратных скобок **[ ]**. Например, следующее выражение, использующее операцию **конкатенации**, **al = [1 2 3]** формирует переменную с именем al, являющуюся одномерным массивом из трех элементов (вещественных чисел). При применении операции конкатенации объединяемые в одномерный массив элементы должны располагаться между открывающей и закрывающей квадратными скобками и отделяться друг от друга либо пробелом, либо запятой. Так что выражение **al = [1, 2, 3]** по своему результату абсолютно идентично предыдущему. Однако если массивы состоят из комплексных чисел или элементы задаются выражениями, то с точки зрения наглядности лучше использовать в качестве *разделителя элементов* запятую, как в следующем примере, в котором создается массив комплексных чисел.

d = [l+2i, 2+3i, 3-7i];

Для доступа к индивидуальному элементу одномерного массива нужно применить *операцию индексации,* для чего после его имени указать в круглых скобках индекс (т. е. номер) элемента. В итоге третий элемент массива **al** обозначается как **al (3)**, первый – как **al (1)**, второй – как **al (2)**.

Если требуется изменить третий элемент сформированного операцией конкатенации массива **al**, то можно применить операцию индексации и операцию присваивания.

al(3) =789

Количество элементов в одномерном массиве всегда можно узнать с помощью функции **length**.

length(al)

ans = 3

Теперь создадим одномерный массив **a2** без применения операции конкатенации. Используем другой способ: будем прописывать каждый элемент создаваемого массива по-отдельности.

a2(1) =67

a2(2) = 7.8

a2(3) = 0.017

Такое постепенное создание массива из трех элементов возможно потому, что система MATLAB с каждым новым присваиванием автоматически перестраивает свою служебную информацию о массиве, а также область памяти, от-

водимую под данные (под элементы). После первого присваивания MATLAB считает, что массив **a2** состоит из одного элемента. При втором присваивании выясняется, что есть еще и второй элемент. Тут уже система MATLAB вынуждена перестраивать структуру памяти, отведенную под данный массив.

Теперь вернемся к рассмотрению различных способов создания одномерных массивов. Еще один способ основан на применении специальной операции, обозначаемой двоеточием. Эту операцию можно назвать операцией формирования диапазона числовых значений. Пусть требуется сформировать одномерный массив чисел в диапазоне от 3.7 до 8.947 с приращением 0.3. Легче всего решить эту задачу с помощью операции двоеточие.

diapl = 3.7 : 0.3 : 8.947;

Операция формирования диапазона работает следующим образом. Сначала она включает в формируемый массив *левую границу диапазона* (это число, стоящее левее первого двоеточия). Затем она к этому числовому значению прибавляет *приращение,* которое указывается после первого двоеточия. Если сумма не превосходит *верхней границы диапазона* (число, стоящее после второго двоеточия), то она включается в качестве элемента в формируемый массив. Это все повторяется до тех пор, пока очередное числовое значение не превысит верхнюю границу.

Несмотря на подробное объяснение работы данной операции, довольно трудно в уме подсчитать количество попадающих в заданный диапазон и, соответственно, в массив **diapl** элементов.

Поэтому лучше вызвать функцию **length (diap1) ans = 18** и выяснить, что в сформированный с помощью операции двоеточие массив **diapl** попало 18 элементов.

Чаще всего эту операцию применяют для формирования диапазона целых числовых значений.

diap2 =4:2: 26;

Если приращение равно единице, то его можно для краткости опустить.

diap3 =2:45;

В случае целых чисел количество попадающих в заданный диапазон элементов формируемого массива подсчитывается без всякого труда. Совершенно очевидно, что в массив **diap3** попадает 44 элемента.

Двумерные массивы можно трактовать как наборы чисел, упорядоченные в виде *прямоугольной таблицы.* Для доступа к индивидуальному элементу используются два индекса: номер строки и номер столбца (на пересечении которых и стоит выбранный элемент). Говорят, что двумерные массивы имеют размерность, равную двум.

Двумерный массив характеризуется количеством строк и количеством столбцов (это размеры вдоль каждой из размерностей). Сформируем операцией конкатенации двумерный массив **A**, состоящий из двух столбцов и трех строк (рисунок 2.2).

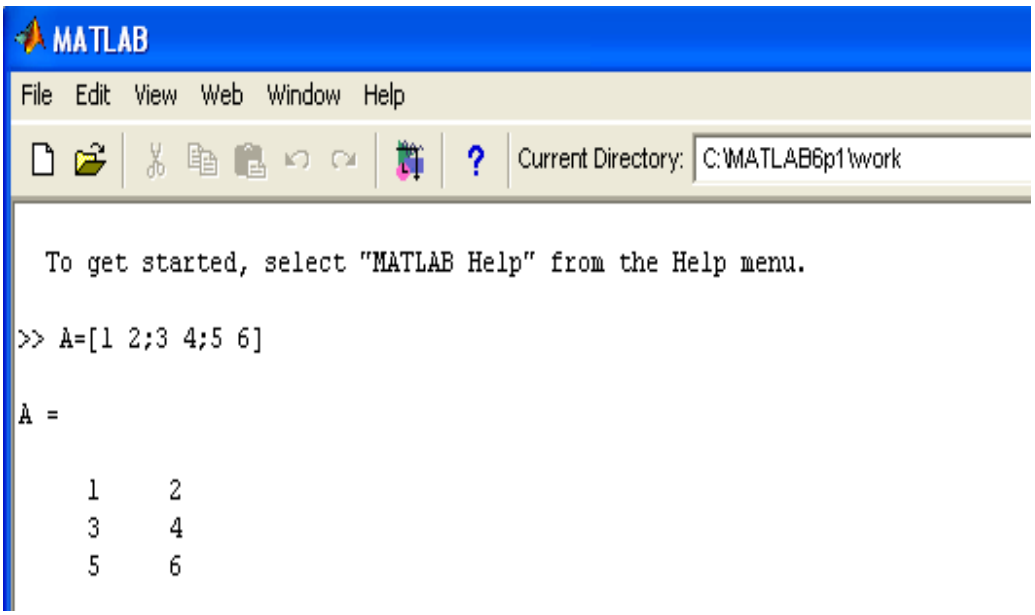


Рисунок 2.2

Из рисунка видно, что в качестве *разделителя строк* в формируемом с помощью операции конкатенации двумерном массиве применяется *точка с запятой.* Это еще одно предназначение точки с запятой в М-языке системы MATLAB.

Чтобы узнать *размеры* двумерного массива, нужно использовать функцию **size**. Для двумерного массива **A** получается следующий результат:

size(A)
ans =
3 2

где первым показывается число строк, а вторым – число столбцов.

Структуру созданных массивов можно также узнать с помощью команды **whos**, которая работает со всеми переменными из текущего рабочего пространства системы MATLAB.

Команда **whos** для всех переменных из текущего рабочего пространства системы MATLAB показывает размер **(Size)** как количество размерностей и количество элементов по каждой из них. Помимо этого, показывается размер области памяти в байтах **(Bytes)**, отводимый под все элементы массива, а также показывается тип данных элементов массива (пока это только **double array**).

Вычисление функций от массивов

В традиционных языках программирования вычисления с массивами осуществляются поэлементно в том смысле, что нужно запрограммировать каждую отдельную операцию над отдельным элементом массива. В М-языке системы MATLAB допускаются мощные групповые операции над всем массивом сразу. Именно групповые операции системы MATLAB позволяют чрезвычайно компактно задавать выражения, при вычислении которых реально выполняется гигантский объем работы.

В частности, в М-языке системы MATLAB они позволяют производить групповые вычисления над массивами, используя обычные математические функции, которые в традиционных языках программирования работают только со скалярными аргументами. В результате с помощью крайне компактных записей, удобных для ввода с клавиатуры в интерактивном режиме работы с командным окном системы MATLAB, удается произвести большой объем вычислений. Например, всего два коротких выражения

x = 0 : 0.01 : pi/2; y = sin(x);

вычисляют значения функции **sin** сразу в 158 точках, формируя два вектора **x** и **y** со 158 элементами каждый.

Это весьма большой объем информации о функции, достаточный для построения ее подробного графика.

Аналогично можно вычислить и синус от двумерного массива числовых аргументов.

Задание 1

Выполните следующее.

1 Поочередно выполните приведенные в примере команды.

Анализируя результаты выполнения команд, установите возможности обработки массивов

d=0.5:0.3:2.5
d1=.5:.3:2.5
=.5+1:.3-.1:2.5*2
length(d)
d(end)
d(end-2)
d(1)
d(0)
d(2:7)
d(7:-1:2)
d(150)
f=linspace(1.5,30,143);
length(f)

2 Сформируйте вектор-столбец **v1**, используя операцию конкатенации.

3
4
5

3 Сформируйте вектор-столбец **v2**, используя поэлементное присваивание.

6
7
8

4 Используя **v1** и **v2**, сформируйте матрицу **v3**.

3 6
4 7
5 8

5 Определите значения синуса для всех элементов матрицы **v2**.

6 Сформируйте последовательность (вектор) **v3**, используя функцию **linspace().**

1.5000 2.0000 2.5000 3.0000 3.5000 4.0000 4.5000 5.0000 5.5000 6.0000

Аргументы функции, их назначение установите посредством анализа рассмотренного примера применения и с помощью справочной системы.

2.3 Лабораторная работа № 8. Операции с векторами и матрицами в системе MATLAB

Цель: изучить реализацию средствами системы MATLAB основных операций с векторами и матрицами.

Теоретические сведения

MATLAB – система, специально предназначенная для проведения сложных вычислений с векторами, матрицами и многочленами. При вводе значения векторов и матриц перечисляются в квадратных скобках. Для разделения столбцов используются пробелы, для разделения строк – знак «;».

Возможен ввод элементов векторов и матриц в виде арифметических выражений, содержащих любые доступные системе функции.

Возможно задание векторов и матриц с комплексными элементами.

Матрицы можно расширять, используя матрицы малых размеров как элементы матриц больших размеров.

Генерацию некоторых наиболее распространенных видов матриц обеспечивают следующие матричные функции:

zeros(m,n) – генерация матрицы с нулевыми элементами;

ones(m,n) – генерация матрицы с единичными элементами;

rand(m,n) – генерация матрицы с элементами, имеющими случайные значения;

eye(m,n) –генерация матрицы с единичными диагональными элементами (m – количество строк; n – количество столбцов матрицы);

a(i,j) – выделение элемента i-й строки j-го столбца;

a(i,:) – выделение i-й строки;

a(:,j) – выделение j-го столбца.

В MATLAB возможны следующие операции с векторами и матрицами:

+ – сложение;

– – вычитание;

* – умножение;

\ / – деление;

' – транспонирование;

^ – возведение в степень;

inv(m) –обращение матрицы;

pinv(m) – псевдообращение матрицы;

sqrtm(m) – матричный квадратный корень;

poly(m) – вектор с коэффициентами характеристического многочлена матрицы;

det(m) –значение определителя матрицы;

trace(m) – след матрицы;

rank(m) – ранг матрицы.

Система MATLAB имеет ряд функций, предназначенных для обработки данных, заданных в матричной или векторной форме.

Функция **max(v)** возвращает значение максимального по значению элемента вектора **v**. Если ее аргументом является матрица, например **max(m),** то функция возвращает вектор-строку, содержащий значения максимальных элементов каждого из столбцов.

Аналогично действует функция **min(m)**, выделяющая элементы с минимальными значениями. Функция **mean(v)** возвращает среднее значение элементов вектора **v**, а функция **mean(m)** с матричным аргументом возвращает вектор-строку средних значений каждого из столбцов данных вектора **v**. Функция сортировки **sort(v)** возвращает вектор, элементы которого расположены в порядке роста их значений. Для матричного аргумента эта функция возвращает матрицу, у которой отсортированы элементы каждого столбца.

Функция **sum(v)** возвращает сумму элементов вектора v, а для матричного аргумента функция sum(m) возвращает вектор-строку сумм элементов по каждому из столбцов. Аналогично функция **prod(m)** возвращает вектор произведений элементов каждого из столбцов.

Порядок выполнения лабораторной работы

1 Ввод с клавиатуры векторов и матриц.

Ввести:

– вектор-строку (v);

– вектор-столбец (w);

– матрицу (m).

2 Генерация матриц специального вида.

Создать:

– матрицу с нулевыми элементами (m0);

– матрицу с единичными элементами(m1);

– матрицу с элементами, имеющими случайные значения(mr);

– матрицу с единичными диагональными элементами(me).

Работа с массивами MATLAB

Все данные MATLAB представляет в виде массивов. Очень важно правильно понять, как использовать массивы. Без этого невозможна эффективная работа в MATLAB, в частности, построение графиков, решение задач линейной алгебры, обработки данных, статистики и многих других. В данном подразделе описаны вычисления с векторами.

Важно понять, что вектор, вектор-строка или матрица являются математическими объектами, а одномерные, двумерные или многомерные массивы – способы хранения этих объектов в компьютере. Далее будут использоваться слова *вектор* и *матрица*, если больший интерес представляет сам объект, чем способ его хранения. Вектор может быть записан в столбик (вектор-столбец) и в строку (вектор-строка).

Ввод, сложение и вычитание векторов

Работу с массивами начнем с простого примера – вычисления суммы векторов. Для хранения векторов используйте массивы **a** и **b**.

Задание 1

Введите массив **a** в командной строке, используя квадратные скобки и разделяя элементы вектора точкой с запятой:

» a = [1.3; 5.4; 6.9]

a =

1.3000

5.4000

6.9000

Так как введенное выражение не завершено точкой с запятой, то MATLAB автоматически вывел значение переменной **a**. Введите теперь второй вектор, подавив вывод на экран.

» b = [7.1; 3.5; 8.2];

Для нахождения суммы векторов используется знак ‹‹+››.

Вычислите сумму, запишите результат в массив **c** и выведите его элементы в командное окно.

Узнайте размерность и размер массива **a** при помощи встроенных функций **ndims** и **size,** сохраните вычисленные значения, выбрав имена переменных самостоятельно.

Примечание – Если размеры векторов, к которым применяется сложение или вычитание, не совпадают, то выдается сообщение об ошибке.

Задание 2

Введите две вектор-строки:

» v1 = [2 -3 4 1];

» v2 = [7 5 -6 9];

Сделав соответствующие преобразования, найдите произведение этих векторов по правилам линейной алгебры.

Поэлементно умножьте векторы (v3).

Используя поэлементное возведение в степень, возведите вектор v1 во вторую степень, а также принимая в качестве показателя степени вектор v2 (v4).

Показателем степени может быть вектор той же длины, что и возводимый в степень. При этом каждый элемент первого вектора возводится в степень, равную соответствующему элементу второго вектора.

Выполните поэлементное деление и обратное деление векторов (v5, v6).

Задание 3

Используя ранее введенные векторы v1 и v2, формируйте новые массивы v7 и v8, применяя операцию конкатенации и проведя соответствующие преобразования векторов.

```
V7=
2    -3     4     1
7     5    -6     9
V8=
2      7
-3     5
4     -6
1      9
```

Задание 4

Для помещения определенных элементов вектора в другой вектор в заданном порядке служит индексация при помощи вектора. Запись в массив w четвертого, второго и пятого элементов v производится следующим образом:

```
» v= [23 45 11 99 33]
» ind = [4 2 5];
» w = v(ind)
w =
99  45  33
```

Выполните выборку элементов массива v4(2,1), v4(2,2), v4(4,1), v4(4,2) и запишите их в массив v9.

Составьте массив w2, содержащий элементы w, кроме четвертого. В этом случае удобно использовать двоеточие и сцепление строк:

```
» w2 = [w(l:3)  w(5:7)]
w2 =
0.1000  2.9000  3.3000  2.6000  7.1000  9.8000
```

Элементы массива могут входить в выражения. Нахождение, например, среднего геометрического из элементов массива **u** можно выполнить следующим образом:

```
» gm = (u(l)*u(2)*u(3))^(l/3)
gm =
17.4779
R1 =
9.4000  7.1000  1.3000  0.8000  -2.3000  -5.2000
```

Упорядочение элементов в порядке возрастания их модулей производится с привлечением функции **abs**:

```
» R2 = sort(abs(R1))
R2 =
0.8000  1.3000  2.3000  5.2000  7.1000  9.4000
```

Вызов **sort** с двумя выходными аргументами приводит к образованию массива индексов соответствия элементов упорядоченного и исходного массивов:

```
» [rs, ind] = sort(r)
```

```
rs =
-5.2000  -2.3000  0.8000  1.3000  7.1000  9.4000
ind =
3    2    5    6    4    1
```

2.4 Лабораторная работа № 9. Построение графиков

Цель: освоить способы построения и оформления двумерных и трехмерных графиков.

Задание 1

Задана функция $y(x)=e^{-x}\sin 10x$ на отрезке [0, 1] с шагом 0.1.

Постройте график функции. Для этого введите в командной строке

>> **plot(x,y).**

После выполнения команды на экране появится окно **Figure No. 1** с графиком функции. Команда **plot** соединяет точки с координатами **(x(i), y(i))** прямыми линиями, автоматически масштабируя оси для оптимального расположения графика в окне. Построенный график не должен иметь изломов, так как сама функция гладкая. Для более точного представления функции нужно задать большее количество точек на отрезке [0,1].

Задание 2

1 При помощи клавиш ↑ ↓ найдите ранее введенную команду задания вектора **x**, задайте более мелкий шаг (0.01), найдите и выполните команду вычисления функции и заново запустите построение графика. У вас должен получиться гладкий график функций.

Таким образом, подбирая шаг изменения аргумента, можно добиться точного представления функции на графике.

2 Графическая функция **fplot** строит график функции без задания пользователем фиксированного шага изменения аргумента.

Функция задается в символьном виде

Синтаксис функции

fplot ('f(x)', [xmin xmax])

Строит график функции **f(x)** в интервале изменения аргумента **x** от **xmin** до **xmax.**

Постройте график функции **$y(x)=e^{-x}\sin 10x$**, используя ***fplot.***

Дополните график сеткой.

Включение и выключение сетки

В математической, физической и иной литературе при построении графиков в дополнение к разметке осей часто используют масштабную сетку. Команды **grid** позволяют задавать построение сетки или отменять это построение:

grid on — добавляет сетку к текущему графику;

grid off — отключает сетку;

grid — последовательно производит включение и отключение сетки.

Задание 3

Наложение графиков друг на друга.

1 Во многих случаях желательно построение наложенных друг на друга графиков в одном и том же окне. Для этого служит команда продолжения графических построений hold. Она используется в следующих формах:

hold on – обеспечивает продолжение вывода графиков в текущее окно, что позволяет добавлять последующие графики к уже существующим;

hold off – отменяет режим продолжения графических построений;

hold – работает как переключатель, последовательно включая режим продолжения графических построений и отменяя его.

Используя команду **hold**, постройте в одном окне графики заданных функций.

Диапазон изменения x

» x=-5:0.1:5;

Функции: **x.sin(x)**, **sin(x) *cos(x)**, **2*sin(x).cos(x)**, **4*sin(x) *.cos(x).**

На рисунке. 2.3 отображены полученные в одном окне графики четырех приведенных функций.

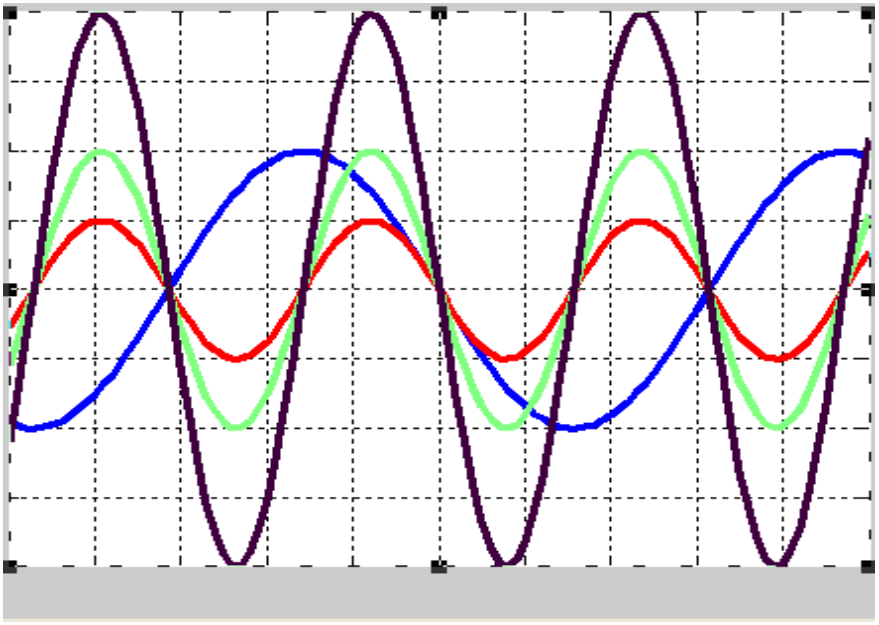


Рисунок 2.3

Проведено дополнительно форматирование – изменены толщина и цвет линий.

2 Постройте на отрезке [-1,-0.3] графики функций **f(x)=sin(1/x^2)** и **g(x) = sin(1.2/x^2)**, используя команду **plot(x, f, x, g)**.

Команда **plot** позволяет задать стиль и цвет линий, например, **plot(x,f,'k-',x,g,'k:')** строит первый график сплошной черной линией, а второй – пунктирной черной линией (рисунок справа). Стили задаются в виде набора трех символов, заключенных в одиночные кавычки (таблица 2.1). Порядок следования символов неважен.

Таблица 2.1

Цвет	Тип точки	Тип линии
b синий	**.** точка	- сплошная
g зеленый	**o** кружок	**:** пунктирная
r красный	**x** крестик	-. штрихпунктирная
c голубой	+ плюс	-- штриховая
m фиолетовый	* звездочка	
y желтый	**s** квадрат	
k черный	**d** ромб	

3 Если одновременно нужно визуализировать несколько графиков в разных окнах, достаточно выполнить команды:

>> w=exp(x)

>>figure;plot(x,w) Команда **figure** создаст новое графическое окно.

Задание 4

Используя функцию **subplot**, постройте:

– графики функций y=sin(x); z=cos(x); w=exp(x);

– графики функций y(x) и z(x) – в одном подокне,

– график функции w=exp(x) – в другом подокне.

Вектор **x=[0:0.01:2];**

Функция **subplot(m,n,p)** используется для указания подобласти, в которой после ее применения с использованием **plot()** или **fplot()** будут построены графики.

У функции **subplot(m,n,p)** три аргумента:

m – число рядов подобластей;

n – число колонок подобластей;

p – номер подобласти (по нарастанию с переходом на следующий ряд).

Задание 5

Чтобы построить график в полярных координатах, надо использовать команду polar. Например, для построения графика функции r= sin(3φ) в полярных координатах следует выполнить группу команд:

>> phi=0:0.01:2*pi; r=sin(3*phi);

>> polar(phi,r)

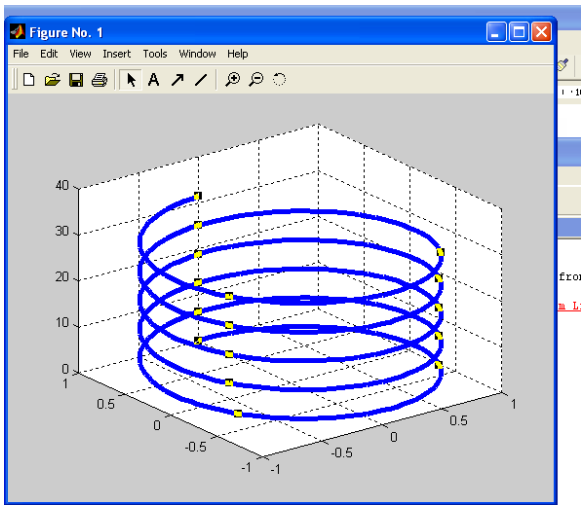
Постройте график функции r1=sin(φ)*cos(φ) для φ ∈pi* [-2, 2] в полярной системе координат.

Трехмерная графика

Каждая точка в пространстве характеризуется тремя координатами. Набор точек, принадлежащих одной линии в пространстве, нужно задать в виде трех векторов (первый должен содержать первые координаты этих точек, второй – вторые, третий – третьи). Вот эти три вектора и подаются на вход функции **plot3**, которая осуществит проектирование соответствующей трехмерной линии на

плоскость и построит результирующее изображение. Так, например, следующий фрагмент кода построит изображение, представленное на рисунке 2.4.

Выполните в командном окне:




```
>> t=0:pi/50:10*pi;
>> x=sin(t);
>> y=cos(t);
>> plot3(x,y,t);
>> grid on
```

Рисунок 2.4

Функция **plot3** в определенном смысле является аналогом функции **plot**. С помощью **plot3** формируется построение линии в трехмерном пространстве по заданным трем векторам.

Задание 6

Построение трехмерной пространственной спирали.

```
t=0:0.05:9*pi;
x=2*sin(t);
y=cos(t);
% t, x, y – векторы одинакового размера
plot3(x,y,t,'r*'),grid,
xlabel('axis X'),ylabel(' axis Y'),zlabel(' axis Z-t')
title('Spatial spiral)
```

Выполните следующее.

1 Сохраните программу в рабочей папке, например, под именем **sp3**.
2 Поменяйте местами **x, y, t** в функции **plot3**.
3 Смените начертание и цвет графика.
4 Установите следующий диапазон для t: t=-9*pi:0.05:9*pi.

Задание 7

Построение сферы по окружностям.

```
n=input('n=');   % Клавиатурный ввод числа n по запросу в командном окне
t1=pi*(-n:5:n)/n;
t2=(pi/2)*(-n:5:n)'/n;     % транспонированный вектор
X=cos(t2)*cos(t1);
Y=cos(t2)*sin(t1);
```

```
E=ones(size(t1));   % матрица единиц размерности вектора t1
Z=sin(t2)*E;
plot3(X,Y,Z,'r'),grid,title('Sphere')
```

Выполните следующее.

1 Сохраните программу в рабочей папке под именем sfera3.

2 Программу выполните при различных значениях **n.**

3 При **n =100** измените шаг в массивах **t1 и t2: для t1 и t2** одновременно: **3, 1, 10, 25, 50**.

4 Для t1=50, для t2=1; для t1=1, для t2=50.

5 Программу **sfera3** выполните без клавиатурного ввода для заданного числа **n.**

6 Установите в программе различные цвета изображения сферы.

Построение пространственных сетчатых фигур – mesh

Задание 8

Формирование прямоугольной сетки на плоскости – **meshgrid**.

```
» [x,y]=meshgrid(-5:0.5:5,-5:0.5:5);
» plot(x,y), xlabel('X'), ylabel('Y')
```

Результатом действия функции **meshgrid** является формирование «основания» в плоскости XOY для построения над ним пространственной фигуры.

```
[x,y]=meshgrid(-5:0.1:5,-5:0.1:5);
Z = 1*x.* exp(-x.^2 - y.^2);
mesh(Z), xlabel('X'), ylabel('Y'), zlabel('Z'), title('Z-surface')
```

Выполните следующее.

1 Сохраните программу в рабочей папке под именем mesh1.

2 Коэффициент 1 поочередно замените: 2, 5, 10, 20.

3 Для сравнения примените plot3(x,y,Z), grid вместо mesh(Z). Файл сохраните с именем mesh1_1.

Задание 9

```
[x,y]=meshgrid(-15:0.2:15,-15:0.2:15);
R=sqrt(x.^2+y.^2)+0.001;
%Коэфф. 0.001 введен для исключения деления на нуль
Z=1*sin(R)./R;
mesh(Z)
```

Выполните следующее.

1 Сохраните программу в рабочей папке под именем mesh2.

2 Коэффициент 1 поочередно замените: 5, 10, 20, 0.5, 0.1.

3 Для сравнения примените plot3(x,y,Z), grid вместо **mesh(Z).** Файл сохраните с именем mesh2_1.

2.5 Лабораторная работа № 10. Встроенные средства решения типовых задач алгебры и анализа

Цель: научиться применять специальные функции для решения задач алгебры и анализа.

В лабораторной работе приводятся основные теоретические сведения о рассматриваемых задачах и функциях MATLAB и примеры их использования.

Рекомендуется примеры выполнить, вводя информацию в командном окне Matlab. Примеры можно использовать в качестве шаблонов при выполнении индивидуальных заданий.

Матричные вычисления в MATLAB

Все переменные в **MATLAB** представляют собой матрицы. Матрицы могут быть заданы различными способами:

– введены явно с помощью списка элементов;

– сгенерированы встроенными операторами или функциями;

– созданы в m-файлах;

– загружены из внешнего файла данных.

Ввод больших матриц обычно выполняется с помощью m-файлов, так как в них легче находить и исправлять ошибки.

В системе **MATLAB** существуют встроенные функции для создания матриц:

zeros – создает матрицу, заполненную нулями;

ones – создает матрицу, заполненную единицами;

eye – создает нулевую матрицу с диагональю, заполненную единицами;

rand – создает матрицу, заполненную случайными числами с равномерным распределением;

randn – создает матрицу, заполненную случайными числами с нормальным распределением.

Для работы с матрицами в системе **MATLAB** существует множество функций. Например:

det – определитель матрицы;

trace – суммирует диагональные элементы;

rank – вычисляет ранг матрицы.

Не следует забывать, что и все арифметические операторы **(+, -, *, /, ^)** являются матричными.

Так, например, решением системы линейных алгебраических уравнений, заданной в матричной форме **X*A = B**, является вектор-столбец **X = B / A**. Существуют также специфично матричные операторы, например, транспонирование – «‘» или обратное деление – «\». Обратное деление позволяет найти решение матричного уравнения **A*X = B** как **X = A\B**.

Функции и операции по обработке матриц

В системе MATLAB основной единицей данных является матрица, поэтому система имеет обширный набор стандартных функций и операций по обработке матриц, который позволяет:

– формировать новые матрицы стандартного вида;

– выполнять матричные арифметические операции;

– вычислять матричные характеристики и математические функции.

Для формирования новых матриц стандартного вида применяются следующие системные функции.

rand(M,N) – формирует прямоугольную матрицу размерностью M×N, элементами которой являются случайные числа в интервале (0.0; 1.0), функция **rand** без параметров формирует одно случайное число в том же интервале.

ones(M,N) – формирует единичную матрицу размерностью M×N.

zeros(M,N) – формирует матрицу размерностью M×N, состоящую из нулей.

diag(V) – создает диагональную матрицу, в которой элементы вектора V являются элементами главной диагонали.

Матричные арифметические операции следующие.

A+B , A-B – матричное сложение и вычитание. Оба операнда этой операции должны иметь одинаковую размерность, если они являются матрицами. Один из операндов может быть скалярной величиной.

A*B – матричное умножение. Операция выполняется по правилам матричного умножения. Число столбцов матрицы **A** должно быть равно числу строк матрицы **B**.

A \ B – левое деление матриц. Осуществляет решение системы линейных алгебраических уравнений A*X=B. Число столбцов матрицы A должно быть равно числу строк матрицы B.

A / B – правое деление матриц. Осуществляет решение системы линейных алгебраических уравнений X*A=B.

X ^ P – возведение матрицы в степень. Эта операция при скалярном значении **P** возводит квадратную матрицу **X** в степень **P**. Если **X** – скалярная величина, а **P** – квадратная матрица, то **X^P** возводит X в матричную степень **P**. Данная операция является ошибочной, если оба операнда – матрицы.

В MATLAB существуют матричные операции, которые выполняются над каждым элементом матрицы. Это такие операции, как:

.* – поэлементное матричное умножение;

. – поэлементное левое деление матриц;

. / – поэлементное правое деление матриц;

.^ – поэлементное возведение матрицы в степень.

Оба операнда этих операций должны иметь одинаковую размерность или один из них должен являться скалярной величиной.

Матрицы могут быть аргументами стандартных математических функций системы, применяемых не к отдельным элементам матрицы, а к матрице целиком.

Решение уравнений

1 Для решения уравнения Y(x)=0, где Y(x) является полиномом, используется стандартная функция **roots** следующего общего вида:

roots(a),

где **a** – вектор коэффициентов перед неизвестными полинома размерностью n+1(n – порядок полинома).

Результатом работы этой функции будет вектор корней полинома размерностью **n**. Далее приведены примеры поиска корней полиномиальных уравнений.

Пример 1 – Решить уравнение $3x^3 + x^2 – 10x – 8 = 0$.
В командном окне формируем вектор коэффициентов перед неизвестными
>>v=[3, 1, -10, -8].
Используя функцию roots, находим решение уравнения
>> R1=roots(v)
R1 =
2.0000
-1.3333
-1.0000

Пример 2 – Решение уравнения $x^2 – 5x – 3 = 0$.
>>R2= roots([1, -5, -3])
R2 =
5.5414
-0.5414

2 Для решения нелинейного уравнения f(x) = 0 используется стандартная функция **fzero** следующего упрощенного общего вида:

fzero(f, x0),

где **f** – имя функции f(x) исходного уравнения, **x0** – начальное приближение корня. Если начальное приближение корня – скалярная величина, то результатом работы функции является один вещественный корень.

Если начальное приближение – массив или диапазон, то результат – вектор со множеством корней. Далее приведены примеры использования функции **fzero.**

Пример 3 – Решение уравнения cos(x) – 0.1x = 0.
>> R3 =fzero('cos(x)-0.1*x',1)
R3 = 1.4276

Пример 4 – Решение уравнения sin(x) – 0.5 = 0 в диапазоне x.
>> x0=2;
>>y=inline('sin(x)-0.5');
>>R4=fzero(y,x0)
R4 = 2.6180

Особенность **fzero** заключается в том, что она не вычисляет корни, в которых функция не меняет знак. Многомерным аналогом **fzero** является команда **fsolve,** предназначенная для решения нелинейных уравнений и систем нелинейных уравнений. Одна из модификаций **fsolve** имеет вид **fsolve('file', x0),** где **file –** имя файл функции , вычисляющей вектор-столбец левых частей системы уравнений; **x0 –** вектор-столбец начальных приближений.

Пример 5 – Решить систему нелинейных уравнений

$$\begin{cases} x^2 + y - 3 = 0; \\ y^2 + x - 2 = 0. \end{cases}$$

Решение системы

Возникают следующие вопросы.

1 Имеет ли система вещественные решения?

2 Если решения есть, то как определить начальные приближения?

Ответ на эти вопросы найдем графическим способом.

На рисунке 2.5 приведено графическое решение исследуемой системы нелинейных уравнений.

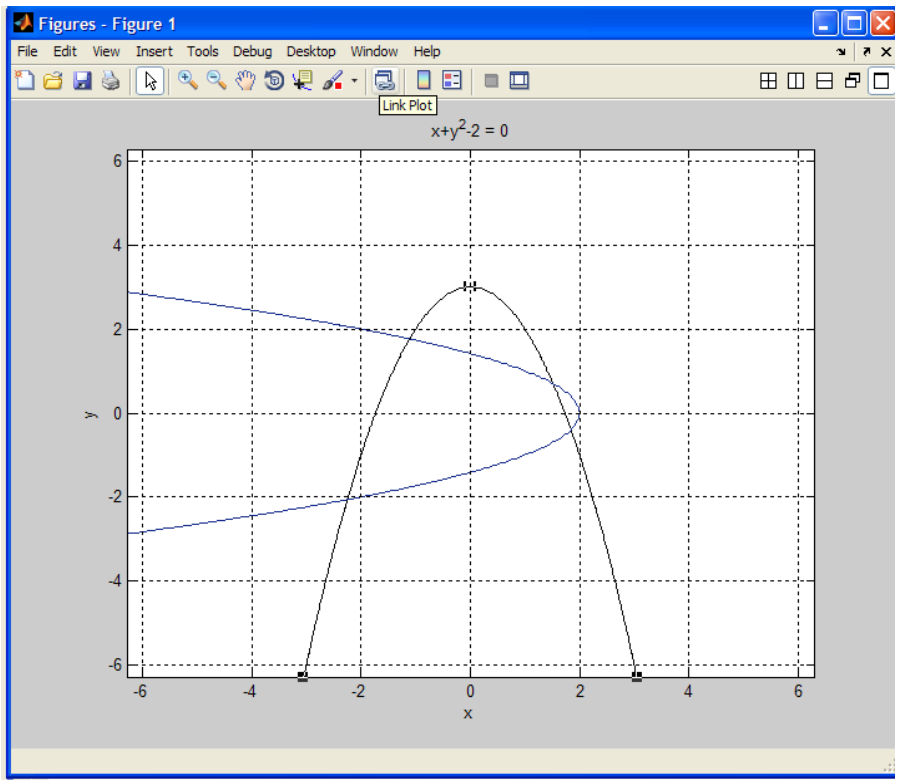


Рисунок 2.5

Как видно, система имеет четыре вещественных решения, а одно из них имеет начальное приближение (-1;2).

Создадим файл-функцию **m_fsolve**, вычисляющую вектор-столбец левых частей системы уравнений:

function F= **m_fsolve(x)**
F=[x(1)^2+x(2)-3; x(2)^2+x(1)-2];
Программа и результаты решения системы уравнений имеют вид:
x0=[-1;2];
X=fsolve('m_fsolve',x0)
X =
-1.1117
1.7640
Получить одновременно приближенное решение и значения левых частей системы уравнений при подстановке в них решения позволяет вызов **fsolve** с двумя выходными аргументами:
[X,f]= fsolve(@m_fsolve,x0)X =
-1.1117
1.7640
f =
1.0e-008 *
0.0445
0.2242
Таким образом, получено одно из приближенных значений. Изменяя стартовые условия, можно найти и остальные вещественные решения.

Список литературы

1 **Кузин, А. В.** Базы данных : учебное пособие / А. В. Кузин, С. В. Левонисова. – 6-е изд., стер. – Москва : Академия, 2016. – 320 с.

2 **Советов, Б. Я.** Базы данных: теория и практика : учебник для бакалавров / Б. Я. Советов. – 2-е изд. – Москва : Юрайт, 2012. – 463 с.

3 **Мартынов, Н. Н.** Matlab 7. Элементарное введение / Н. Н. Мартынов – Москва: КУДИЦ-ОБРАЗ, 2005. – 416 с.

4 **Дьяконов, В. П.** Matlab: учебный курс / В. П. Дьяконов. – Санкт-Петербург: Питер, 2001. – 560 с.

5 Информатика. Базовый курс : учебное пособие / Под ред. С. В. Симоновича. – 3-е изд. – Санкт-Петербург : Питер, 2017. – 640 с. : ил.

6 ЦИТМ Экспонента – официальный дистрибьютор Math Works на территории России и СНГ [Электронный ресурс]. – Режим доступа: http://matlab.exponenta.ru/. – Дата доступа: 10.11.2018.