

Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Двоеглазов Семен Иванович
Должность: Директор
Дата подписания: 30.06.2025 15:30:50
Уникальный программный ключ:
2cc3f5fd1c09cc1a69668dd98bc3717111a1a535



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
СТАРООСКОЛЬСКИЙ ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ
ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГЕОЛОГОРАЗВЕДОЧНЫЙ
УНИВЕРСИТЕТ имени СЕРГО ОРДЖОНИКИДЗЕ»
СОФ МГРИ

Т.В. ИВАНОВА

ИНФОРМАТИКА

Учебное пособие
для выполнения контрольных работ, домашних заданий
студентам 1,2 курсов всех форм обучения

Рекомендовано Ученым советом СОФ МГРИ

Старый Оскол, 2020

УДК 004
ББК 32.988-5

Составитель канд.пед. наук Т.В. Иванова

Рецензент доц., канд. физ.-мат. наук СТИ НИТУ «МИСиС» А.К. Ефимов

Иванова Т. В.

Информатика: учебное пособие для выполнения контрольных работ, домашних заданий студентам 1,2 курсов всех форм обучения/ Т.В. Иванова /Старооскольский филиал ФГБОУ ВО «Российский государственный геологоразведочный университет имени Серго Орджоникидзе», 2020. - 51 с. Утверждено и рекомендовано к изданию Ученым советом Старооскольского филиала ФГБОУ ВО «Российский государственный геологоразведочный университет имени Серго Орджоникидзе» (протокол №11 от 31 августа 2020 г.).

В данном пособии рассмотрены правила оформления контрольной работы (домашнего задания), приведены примеры решения задач по следующим разделам дисциплины «Информатика»: организация данных в ЭВМ, кодирование информации, системы счисления, арифметические и логические основы построения ЭВМ, прикладное программное обеспечение, основы алгоритмизации и программирования, основы математического моделирования.

Учебное пособие предназначено для студентов 1,2 курсов всех форм обучения.

УДК 004
ББК 32.988-5

СОДЕРЖАНИЕ

	Стр.
Введение	4
1.Правила оформления контрольной работы (домашнего задания)	6
2.Примеры решения задач	7
2.1.Задача №1. (Организация данных в ЭВМ. Кодирование информации. Системы счисления)	7
2.2.Задача №2. (Арифметические и логические основы построения ЭВМ)	8
2.3.Задача №3. (Решение задач в табличном процессоре. Процедуры «Поиск решения», «Подбор параметра», построение графика функции)	9
2.4.Задача №4. (Решение задач в табличном процессоре. Решение системы линейных алгебраических уравнений)	11
2.5.Задача №5. (Решение задач в табличном процессоре. Статистические функции)	12
2.6.Задача №6. (Работа с текстовым процессором)	14
2.7.Задача №7. (Работа с базой данных)	15
2.8.Задача №8. (Математические и инженерные расчеты в специализированных программах)	19
2.9.Основы алгоритмизации и программирования	28
2.10.Задача №9. (Выполнение программы линейной структуры)	40
2.11.Задача №10. (Использование циклов)	40
2.12.Задача №11. (Использование условного оператора)	43
2.13.Задача №12.(Использование одномерных массивов)	44
2.14.Задача №13.(Использование двумерных массивов)	46
2.15.Задача №14. (Использование функций)	46
Литература	50

Введение

Большую роль в изучении дисциплины «Информатика» имеет комплекс контрольных работ (для студентов заочной формы обучения), домашних заданий (для студентов очной формы обучения), главной задачей которого является обучение студентов в процессе их самостоятельной работы за компьютером. В учебном пособии для выполнения контрольных работ (домашних заданий) приводятся краткие теоретические сведения и примеры решения задач.

Задачи предназначены для студентов 1 и 2 курсов всех форм обучения.

В таблице представлены номера задач по дисциплине «Информатика» для разных специальностей и направлений.

Специальность (направление)	Семестр	Количество контрольных работ и примеры решения задач
21.05.02 – «Прикладная геология» - заочная форма обучения	1,2	Контрольная работа №1: задачи №1, №2, №3, №4. Контрольная работа №2: задачи №7, №8, №11, №12
21.05.02 – «Прикладная геология» - очная форма обучения	1,2	Домашнее задание №1: задача №1. Домашнее задание №2: задачи №3, №4, №5. Домашнее задание №3: задачи №10, №12, №14. Домашнее задание №4: №8.
21.05.03 – «Технология геологической разведки» - заочная форма обучения	1,2 (1 курс)	Контрольная работа №1: задачи №1, №2, №6, №7. Контрольная работа №2: задачи №5, №3, №4.
	3,4 (2 курс)	Контрольная работа №3: задачи №9, №10, №12, №13 Контрольная работа №4: задачи №14, примеры задачи №8.
21.05.04 – «Горное дело» все формы обучения	1,2	Контрольная работа: задачи №7, №3, №4, №11, №12
21.03.01 – «Нефтегазовое дело» - заочная форма обучения	1,2	Контрольная работа №1: задачи №7, №3, №4 Контрольная работа №2: №11, №12
38.03.01 – «Экономика» - все формы обучения	1,2	Контрольная работа №1: задачи №7, №3, №4 Контрольная работа №2: №11, №12

Номера вариантов контрольной работы (домашнего задания) представлены в нижеследующей таблице и соответствуют последней цифре в номере зачетной книжки студента.

Последняя цифра в номере зачетной книжки	1	2	3	4	5	6	7	8	9	0
№ варианта	1	2	3	4	5	6	7	8	9	10

1. ПРАВИЛА ОФОРМЛЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ (ДОМАШНЕГО ЗАДАНИЯ)

Общие требования.

Работа должна быть выполнена печатным способом на одной стороне листа белой бумаги формата А4 (210×297мм).

При наборе и форматировании текста в среде текстового процессора следует соблюдать следующие требования:

- шрифт Times New Roman, размер 14pt;
- выравнивание – по ширине;
- междустрочный интервал 1,5;
- автоматический перенос слов;
- размеры полей: левое – 20 мм, правое - 20 мм, : верхнее – 20 мм, нижнее – 20 мм;
- отступ первой строки должен быть одинаковым по всему тексту и равен 1,25 см.

Разрешается использовать компьютерные возможности акцентирования внимания на определенных терминах, формулах, определениях с применением разных шрифтов.

Повреждения листов текстовых документов, пометки и следы не полностью удаленного прежнего текста (графики) не допускаются.

Содержание отчета к решениям задач.

- Номер задачи.
- Постановка задачи по варианту.
- Решение задачи с письменными и графическими пояснениями, скриншотами.

2. ПРИМЕРЫ РЕШЕНИЯ ЗАДАЧ

2.1. Задача №1. (Организация данных в ЭВМ. Кодирование информации. Системы счисления)

Выполнить перевод чисел из одной системы счисления в другую согласно таблице:

Перевести из десятичной системы в двоичную систему счисления:	188
Перевести из десятичной системы в шестнадцатеричную систему счисления:	163
Перевести из двоичной в десятичную систему счисления:	10100100
Перевести из двоичной в восьмеричную систему счисления:	10100100
Перевести из двоичной в шестнадцатеричную систему счисления:	10100100

Выполнить сложение двоичных чисел: $00111011 + 00011110$

Выполнить вычитание двоичных чисел: $0011101 - 11110$

Решение.

1. $188_{10} = 10111100_2$

Разряды	7	6	5	4	3	2	1	0
Значения	1	0	1	0	1	1	1	1

2. $154_{10} = 10011010_2$

Разряды	7	6	5	4	3	2	1	0
Значения	1	0	0	1	1	0	1	0

Разбиваем число на тетрады $1001\ 1010 = 9A$ в 16-ной системе счисления.

3. $10101110_2 = 2^7 + 2^5 + 2^3 + 2^2 + 2^1 + 2^0 = 128 + 32 + 8 + 4 + 2 + 1 = 175_{10}$

4. Разбиваем число на триады $010\ 101\ 110 = 256$ в 8-ной системе счисления

5. Разбиваем число на тетрады $1010\ 1110 = AE$ в 16-ной системе счисления

6. Сложение двоичных чисел

	110010
+	11001
	1001001

7. Вычитание двоичных чисел

	110010
-	11001
	11001

2.2. Задача №2. (Арифметические и логические основы построения ЭВМ)

Составить таблицу истинности для логического выражения: $F = \neg(A \& B)$.

Основные логические операции.

1. Отрицание $y = \bar{x}$ или $\neg x$

x	y
0	1
1	0

2. Конъюнкция (логическое умножение, логическое И). Является многоместной.

$$y = x_1 \wedge x_0 \text{ или } y = x_1 \& x_0.$$

x_1	x_0	y	$\bar{y}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

3. Дизъюнкция – логическое сложение (логическое ИЛИ).

$$y = x_1 \vee x_0$$

x_1	x_0	y	$\bar{y}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Значения логической функции для разных сочетаний значений входных переменных обычно задаются специальной таблицей. Такая таблица называется **таблицей истинности**.

Приведем таблицу истинности основных логических операций выражения

A	B	$A \& B$	$F = \neg(A \& B)$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

2.3. Задача №3. (Решение задач в табличном процессоре. Процедуры «Поиск решения», «Подбор параметра», построение графика функции)

Будем рассматривать решение задачи в программе MS Excel. Требуется решить задачу двумя способами (с помощью процедуры «Поиск решения», с помощью процедуры «Подбор параметра»).

1 способ.

Используя процедуру **Поиск решения** в MS Excel, найти:

- корень алгебраического или трансцендентного уравнения $f(x)=0$ на отрезке $[a, b]$;
- минимальное и максимальное значение данной функции на отрезке $[a, b]$.

2 способ.

Используя надстройку **Подбор параметра**, найти корень алгебраического или трансцендентного уравнения $f(x)=0$ на отрезке $[a, b]$.

Сравнить полученные значения.

Построить график функции $f(x)$ на отрезке $[a, b]$.

В отчет по выполнению задачи включить для каждого способа решения распечатку окон программы MS Excel с выполняемыми действиями и кратким описанием действий, а также с результатом действий.

Решение.

Используя процедуру **Поиск решения**, найдем корень уравнения $x^4 - x^3 - 2x^2 + 3x - 3 = 0$ на отрезке $[-2; -1]$.

Значение x запишем в ячейку A1, в ячейку B1 напишем формулу для вычисления $f(x)$, выполним команду **Данные–Поиск решения**, затем в открывшемся окне диалога запишем ограничения на значения интервала решения (рис.1).

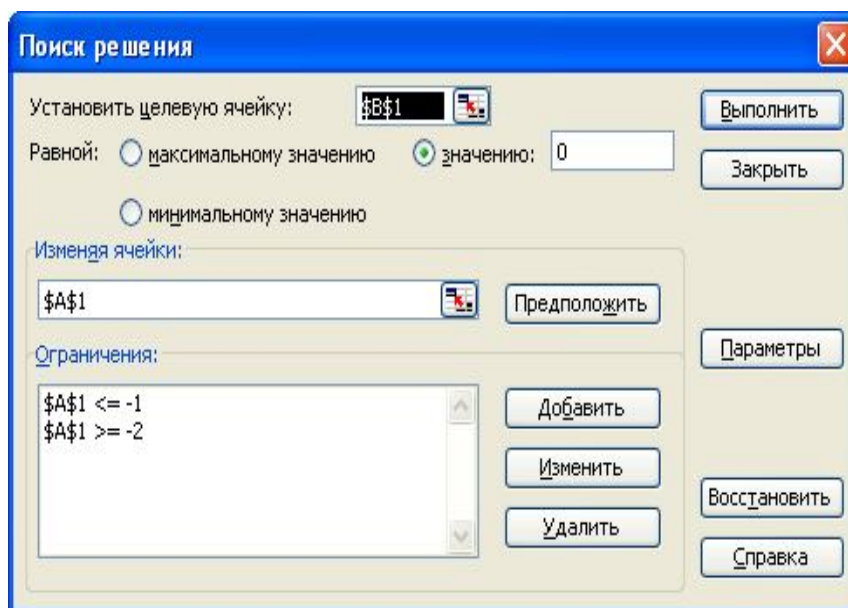


Рис.1. Диалоговое окно Поиск решения.

Нажав кнопку **Выполнить**, найдем значение корня (см. рис. 2).

B1		$=A1^4-A1^3-2*A1^2+3*A1-3$
	A	B
1	-1,732050853	9,10441E-07

Рис. 2. Найденное значение корня уравнения в ячейке A1.

Аналогично, находится максимальное и минимальное значения функции на отрезке $[-2;-1]$ (рис. 3,4).

Рис. 3. Диалоговое окно Поиска максимального значения.

B1		$=A1^4-A1^3-2*A1^2+3*A1-3$
	A	B
1	-2	7,00

Рис. 4. Найденное максимальное значение функции.

Для построения графика функции следует предварительно построить таблицу значений аргумента x и соответствующих им значений функции $f(x)$. Далее с помощью **Мастера диаграмм** можно построить график заданной функции (рис. 5).

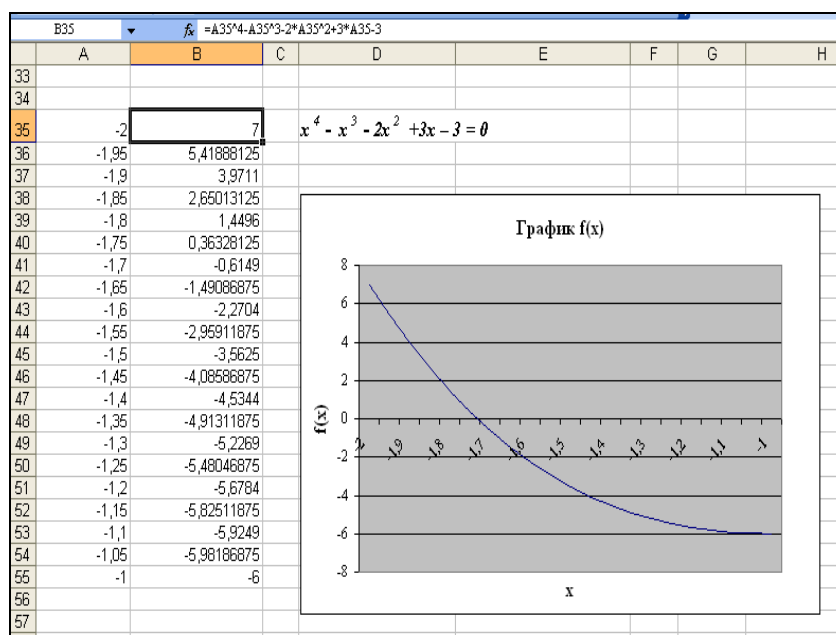
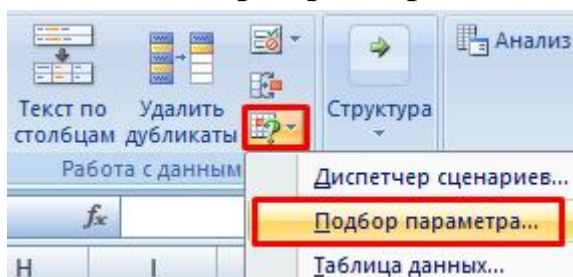


Рис. 5. График функции $f(x)=x^4-x^3-2x^2+3x-3$.

Процедура **Подбор параметра** позволяет в формуле при некоторых известных переменных найти варьируемое значение неизвестного параметра (в данной задаче корень уравнения). Для вызова процедуры необходимо обратиться к пункту меню программы **MS Excel Данные – Анализ «Что-если» - Подбор параметра**.



В результате решения задачи двумя способами должны получить один и тот же ответ, если ответы не совпадают, надо найти ошибку и решить задачу вновь.

2.4. Задача №4. («Решение задач в табличном процессоре. Решение системы линейных алгебраических уравнений»)

Будем рассматривать решение задачи в программе MS Excel.

Постановка задачи. В MS Excel решить систему линейных алгебраических уравнений $Ax=b$ с использованием математических функций для нахождения обратной матрицы A^{-1} и произведения матриц $A^{-1} * b$.

Выполнить проверку найденного решения путем перемножения матриц A и x и сравнения полученного значения с b .

В ячейки рабочего листа программы MS Excel вводится два вида данных: константы и формулы. Константы делятся на числовые, текстовые, логические, типа дата/время. Числа включают цифры и символы-

разделители: десятичная точка, процент, знак числа, денежное обозначение (например, р. или \$). Текст представляет собой последовательность букв, цифр, специальных символов.

Массив – это множество ячеек, содержимое которых обрабатывается как единое целое. Такие ячейки могут указываться как именованный диапазон, например A1:D4 (рис. 6).

	A	B	C	D
1				
2				
3				
4				
5				

Рис. 6. Массив ячеек A1:D4.

Формула массива – это формула, оперирующая с одним или несколькими массивами. Для работы с формулами массива необходимо знать следующее:

- признаком формулы массива являются фигурные скобки в начале и конце формулы;
- фигурные скобки вводятся нажатием комбинации клавиш Ctrl+Shift+Enter;
- при редактировании формул фигурные скобки исчезают, для восстановления фигурных скобок нужно повторно нажать вышеуказанную комбинацию клавиш.

Например, чтобы найти матрицу, обратную к исходной матрице (рис. 7), мы используем формулу массива, примененную к массиву A1:C3.

D1	fx {=МОБР(A1:C3)}					
	A	B	C	D	E	F
1	1	3	4	-1,16667	0,333333	0,5
2	2	3	5	0,5	-1	0,5
3	3	5	6	0,166667	0,666667	-0,5
4						

Рис. 7. Нахождение обратной матрицы.

Аналогично можно перемножать матрицы (см. рис.8), в частности перемножение матрицы D1:F3 на вектор G1:G3.

fx {=МУМНОЖ(D1:F3;G1:G3)}					
C	D	E	F	G	H
4	-1,16667	0,333333	0,5	6	-3,16667
5	0,5	-1	0,5	4	1,5
6	0,166667	0,666667	-0,5	5	1,166667

Рис. 8. Использование формулы массива при перемножении матриц.

2.5. Задача №5. (Решение задач в табличном процессоре. Статистические функции)

Будем рассматривать решение задачи в программе MS Excel.

Одним и тем же вольтметром было измерено n раз напряжение на участке цепи. В результате опытов получены следующие значения напряжения (в вольтах).

Таблица 1

Число измерений	Значения напряжения
26	31,31,32,32,35,33,36,32,34,34,35,32,33,36,37,34,36,35,33,34,36,37,34,39,40,31

Найти среднее значение, выборочные дисперсию и стандартное отклонение и генеральные дисперсию и стандартное отклонение, размах варьирования, моду, медиану, используя функции MS Excel.

1. Занесем исходные данные измерений в ячейки диапазона A2:Z2

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
Результаты измерений	31	31	32	32	35	33	36	32	34	34	35	32	33	36	37

2. Среднее значение равно

B3 $\text{fx} = \text{СРЗНАЧ}(\text{A2:Z2})$

	A	B
1	Результаты измерений	
2		31
3	Среднее значение	34,30769

3. Выборочная дисперсия

B4 $\text{fx} = \text{ДИСПА}(\text{A2:Z2})$

	A	B
1	Результаты измерений	
2		31
3	Среднее значение	34,30769
4	Выборочная дисперсия	5,661538

4. Стандартное отклонение

B5 $\text{fx} = \text{СТАНДОТКЛОНА}(\text{A2:Z2})$

	A	B	C
1	Результаты измерений		
2		31	32
3	Среднее значение	34,30769	
4	Выборочная дисперсия	5,661538	
5	Стандартное отклонение	2,379399	

5. Генеральная дисперсия

B6 $\text{fx} = \text{ДИСПРА}(\text{A2:Z2})$

	A	B
1	Результаты измерений	
2		31
3	Среднее значение	34,30769
4	Выборочная дисперсия	5,661538
5	Стандартное отклонение	2,379399
6	Генеральная дисперсия	5,443787

6. Размах варьирования

B7		fx	$\text{=ABS(МАКС(A2:Z2)-МИН(A2:Z2))}$	
	A	B	C	D
1	Результаты измерений			
2	31	31	32	32
3	Среднее значение	34,30769		
4	Выборочная дисперсия	5,661538		
5	Стандартное отклонение	2,379399		
6	Генеральная дисперсия	5,443787		
7	Размах варьирования	9		

7. Мода

B8		fx	=МОДА(A2:Z2)	
	A	B		
1	Результаты измерений			
2	31	31		
3	Среднее значение	34,30769		
4	Выборочная дисперсия	5,661538		
5	Стандартное отклонение	2,379399		
6	Генеральная дисперсия	5,443787		
7	Размах варьирования	9		
8	Мода	34		
9				

8. Медиана

B9		fx	=МЕДИАНА(A2:Z2)	
	A	B	C	
1	Результаты измерений			
2	31	31		
3	Среднее значение	34,30769		
4	Выборочная дисперсия	5,661538		
5	Стандартное отклонение	2,379399		
6	Генеральная дисперсия	5,443787		
7	Размах варьирования	9		
8	Мода	34		
9	Медиана	34		
10				

2.6. Задача №6 (Работа с текстовым процессором)

Будем рассматривать решение задачи в программе MS Word. В этой задаче задания двух типов.

Задача №6.1. Создание, редактирование и форматирование формул в MS Word.

Методические рекомендации. Для создания формулы в программе MS Word следует вызвать редактор формул **Вставка – Объект – MS Equation** (Office 2003) или **Вставка – Формула** (начиная с Office 2007)

Задача №6.2. Обработка числовой информации в таблицах MS Word.

Методические рекомендации. Правила структуры таблицы MS Word такие же, как и в MS Excel. В примере таблица помещается в диапазон A1:E6.

Результаты тестирования группы ДЭМ-1-1

Фамилия	Правильные ответы			Результат (сумма правильных ответов)
	Тест 1	Тест 2	Тест 3	
Михайлов	16	13	20	
Муравьева	20	14	25	
Щеглов	19	23	16	
Алексеев	14	11	15	

Для вставки расчетной формулы в ячейку E2 таблицы необходимо поместить курсор в эту ячейку, вызвать **Макет – Формула** (начиная с Office 2007) или Таблица – Формула (Office 2003).



В появившемся окне после знака «=» вставить необходимую функцию (SUM – сумма, AVERAGE – среднее, MAX – максимум, MIN – минимум и т.д.). В ячейке E3 должна быть формула = SUM(B3:D3).

2.7. Задача №7. (Работа с базой данных)

Будем рассматривать решение задачи в программе MS Access.

Постановка задачи. Предметная область информационной системы – оптовая база.

Минимальный список характеристик:

Код товара, название товара, количество на складе, стоимость единицы товара, общая стоимость товара.

1. Проектирование базы данных.

Определим типы значений атрибутов и их размер (рис.9):

Код товара – числовой тип, первичный ключ, целое,
название товара – текстовый тип, 25 символов,
количество на складе – числовой тип, целое,
стоимость единицы товара – денежный тип (руб.),
общая стоимость товара - денежный тип (руб.).

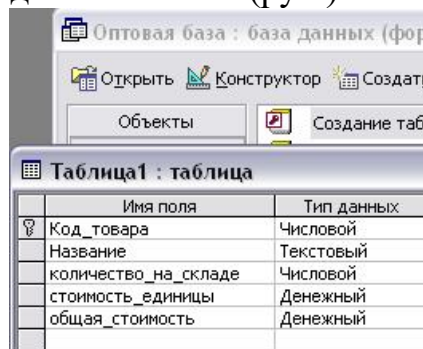


Рис.9. Создание таблицы в режиме Конструктор.

2. Создаем файл в MS ACCESS с именем Оптовая база.mdb.
3. Создаем таблицу с именем Таблица 1 в режиме Конструктор (рис.10).
4. Внесем данные в Таблицу 1.

Таблица1 : таблица					
	Код_товара	Название	количество_на	стоимость_еди	общая_стоимо
▶	1	шариковые ручки	100	2,00р.	200,00р.
	2	цветные карандаши	1000	1,00р.	1 000,00р.
	3	ластик	2000	1,00р.	2 000,00р.
	4	тетрадь в клетку	500	3,00р.	1 500,00р.
	5	тетрадь в линию	1000	3,00р.	3 000,00р.
	6	папка	200	10,00р.	2 000,00р.
	7	обложка универсальная	150	10,00р.	1 500,00р.
*	0		0	0,00р.	0,00р.

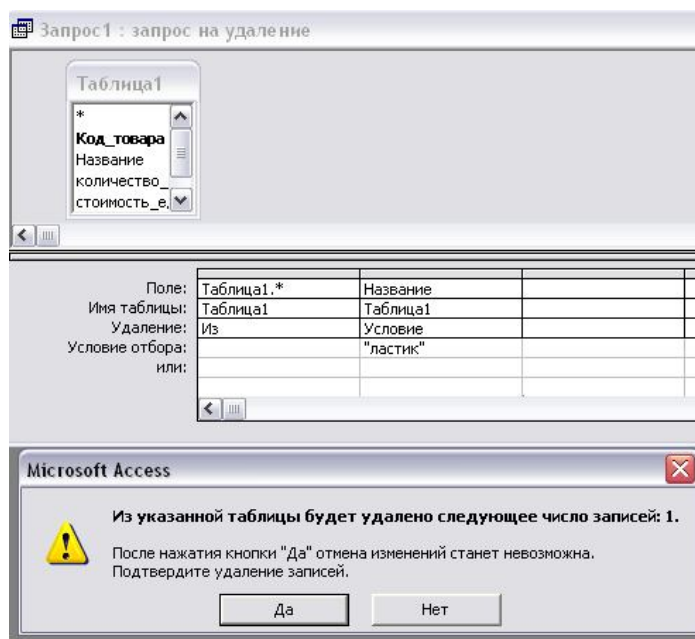
Рис.10. Таблица с исходными данными.

5. Запросы к базе данных.

а. Запрос на удаление данных. Формулировка запроса: удалить из базы данных информацию о товаре с названием «ластик».

Выполним запрос в режиме Конструктор.

Рис.11 Запрос на удаление данных.



Результат: в Таблице 1 удалена строка с названием товара «ластик».

Таблица1 : таблица					
	Код_товара	Название	количество_на	стоимость_еди	общая_стоимо
▶	1	шариковые ручки	100	2,00р.	200,00р.
	2	цветные карандаши	1000	1,00р.	1 000,00р.
	4	тетрадь в клетку	500	3,00р.	1 500,00р.
	5	тетрадь в линию	1000	3,00р.	3 000,00р.
	6	папка	200	10,00р.	2 000,00р.
	7	обложка универсальная	150	10,00р.	1 500,00р.
*	0		0	0,00р.	0,00р.

Рис.12 Результат запроса на удаление данных.

- б. Запрос на обновление данных. Формулировка запроса: изменить название товара для кода 1.

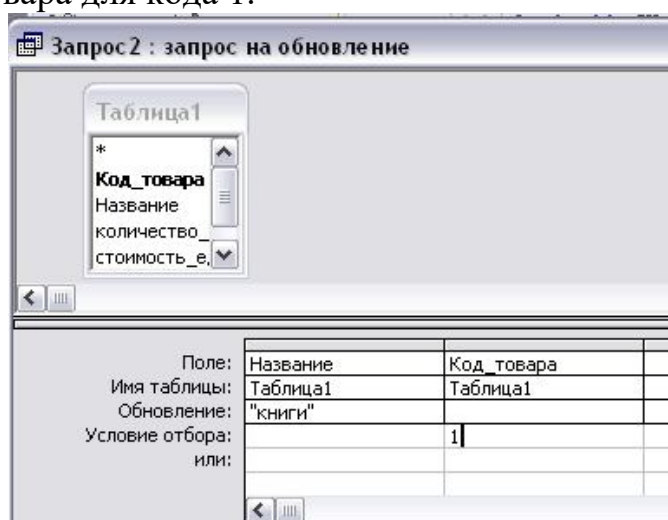


Рис.13. Запрос на обновление данных в режиме Конструктор.

Результат запроса на рис.14. В Таблице 1 для кода товара 1 изменилось название, вместо «шариковые ручки» появилось «книги».

Таблица1 : таблица					
	Код_товара	Название	количество_на	стоимость_еди	общая_стоимо
▶	1	книги	100	2,00р.	200,00р.
	2	цветные карандаши	1000	1,00р.	1 000,00р.
	4	тетрадь в клетку	500	3,00р.	1 500,00р.
	5	тетрадь в линию	1000	3,00р.	3 000,00р.
	6	папка	200	10,00р.	2 000,00р.
	7	обложка универсальная	150	10,00р.	1 500,00р.
*	0		0	0,00р.	0,00р.

Рис.14. Результат запроса на удаление данных.

- с. Запрос на выбор данных по условию. Формулировка запроса: выбрать все сведения о товарах, для которых общая стоимость > 2000 руб.

Запрос3 : запрос на выборку

Таблица1

*
Код_товара
Название
количество_
стоимость_е.

Поле: Таблица1.*
Имя таблицы: Таблица1
Сортировка:
Вывод на экран: ☒
Условие отбора: >2000
или:

Рис.15. Запрос на выбор данных в режиме Конструктор.

Результат.

Запрос3 : запрос на выборку					
	Код_товара	Название	количество_на	стоимость_еди	общая_стоимо
▶	5	тетрадь в линию	1000	3,00р.	3 000,00р.
*	0		0	0,00р.	0,00р.

Рис.16.Результат запроса на выборку.

- d. Запрос на выбор данных с использованием агрегатных функций.
Выбрать наибольшее количество товаров на складе.

Запрос4 : запрос на выборку

Таблица1

*
Код_товара
Название
количество_
стоимость_е.

Поле: количество_на_ск
Имя таблицы: Таблица1
Групповая операция: Max
Сортировка:
Вывод на экран: ☒
Условие отбора:

Рис.17. Запрос на выбор данных в режиме Конструктор.

е. Результат.

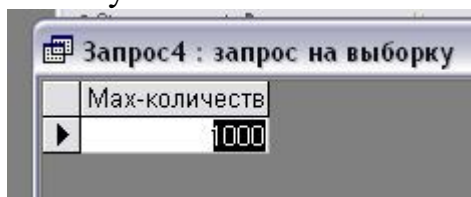


Рис.18.Результат запроса на выбор с использованием функции Max().

2.8. Задача №8. (Математические и инженерные расчеты в специализированных программах)

Примерами специализированных программ являются **MathCAD**, **Scilab**.

Scilab – это система компьютерной математики (СКМ), которая предназначена для выполнения научно-технических расчетов, графической интерпретации полученных результатов и визуального моделирования. Эта система имеет удобный пользовательский интерфейс и развитый язык программирования.

Система имеет несколько режимов работы, каждый из которых поддерживается собственным диалоговым окном (рис. 19):

- командный режим – командное окно;
- программный режим – окно создания и редактирования программных файлов (SCE-файлов);
- графический режим – окно редактирования графиков;
- режим помощи – окно помощи;
- режим демонстрации – окон демонстрационных примеров.

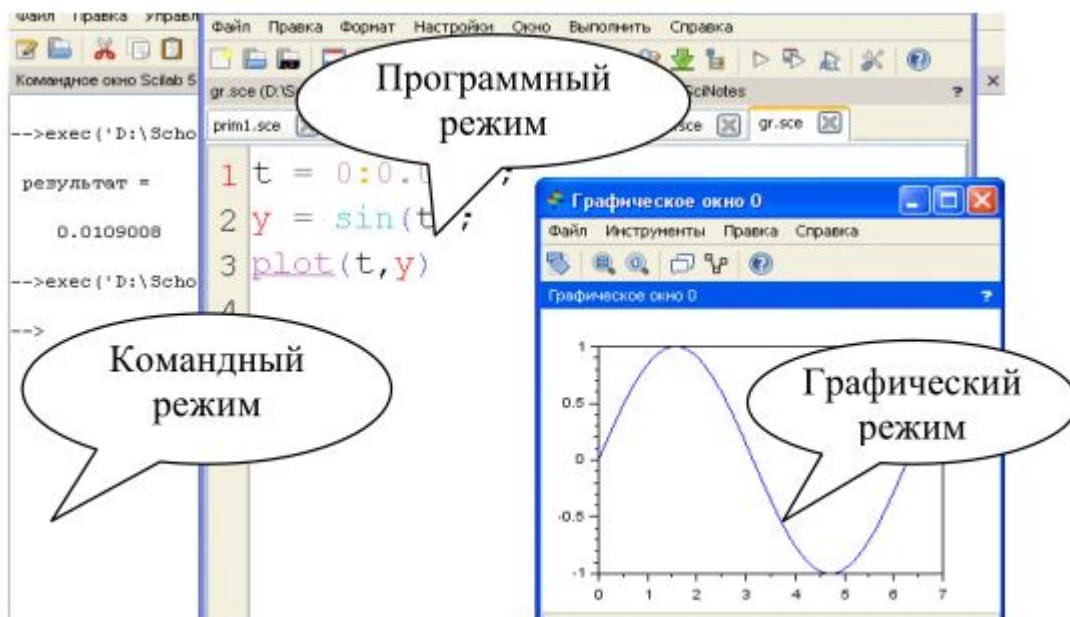


Рис. 19. Режимы работы программы.

При работе в любом из перечисленных режимов могут быть использованы дополнительные информационные окна (рис. 20). Окно

рабочей области (Обозреватель переменных) – предназначено для просмотра и редактирования содержимого рабочей области памяти, в нем указывается имя переменной (массива или структуры), ее размерность и тип.

Окно журнала команд содержит перечень команд, введенных пользователем в командном режиме за текущий и предыдущий сеансы работы с системой.

Окно управления файлами (обозреватель файлов) служит для быстрого доступа к файлам при работе с системой.

Управлять информационными окнами можно с использованием пункта основного меню «Инструменты».

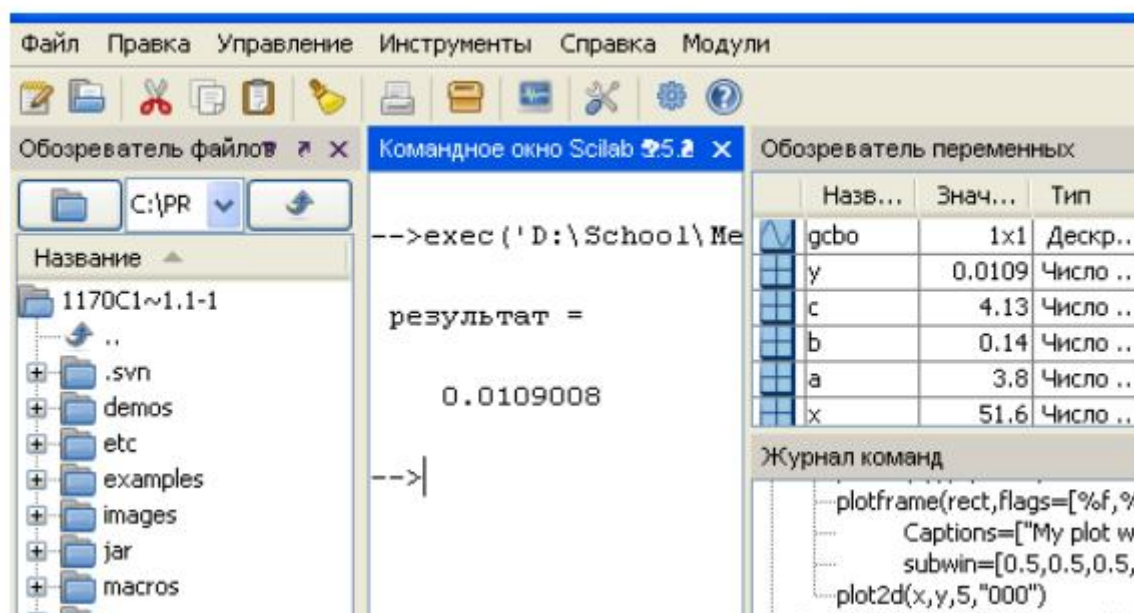


Рис.20. Информационные окна программы.

Создание документа в системе Scilab может выполняться в различных режимах, ниже рассматриваются командный и программный режимы.

Документ при работе в командном режиме представляет собой последовательность команд пользователя и ответов системы, расположенных в командном окне. Символ --> в окне команд показывает, что система готова к диалогу с пользователем. Командная строка может содержать одну или несколько команд, она завершается нажатием клавиши Enter. Строка реакции системы называется строкой вывода, она показывает результаты выполнения команды как значение стандартной переменной ответа ans, либо переменной, заданной пользователем, например:

```
-->5+3
ans=
8
-->b=5+3
b=
8
```

Переменная ans хранится в памяти и может использоваться в дальнейших вычислениях. Выражения или команды в строке разделяются

символами «;» или «;». Результат вычисления выражения или команды, за которой следует запятая, выводится на экран в строку вывода. Результат вычисления выражения или команды, за которой следует точка с запятой, на экран не выводится, но сохраняется в памяти, например:

```
-->x=5,y=x+7;
```

```
x=
```

```
5
```

Отработанная командная строка не может быть выполнена повторно путем возвращения в нее курсора мыши в командном окне. Для нового выполнения команды она должна быть вызвана из стека команд нажатием клавиш $\uparrow$ или $\downarrow$. Если выражение не помещается в одной командной строке, то его можно перенести на следующую строку, а предыдущую закончить тремя точками.

В программном режиме пользователь создает программу, которая состоит из команд и выражений системы Scilab и хранится на диске в виде файла с типом `sce`. Последовательность создания и обработки программы следующая:

Шаг 1. Создать новый файл программы можно с помощью редактора SciNotes командой «Инструменты – Текстовый редактор SciNotes» или первой кнопкой на панели инструментов командного окна.

Шаг 2. Записать файл на диск с именем, содержащим тип `sce`, с помощью команды «Файл – Сохранить как» программного окна.

Шаг 3. Запустить программу на выполнение с помощью меню «Выполнить» программного окна. Командой «Сохранить и выполнить», кнопкой F5 или кнопкой на панели инструментов программного окна (рис. 21).

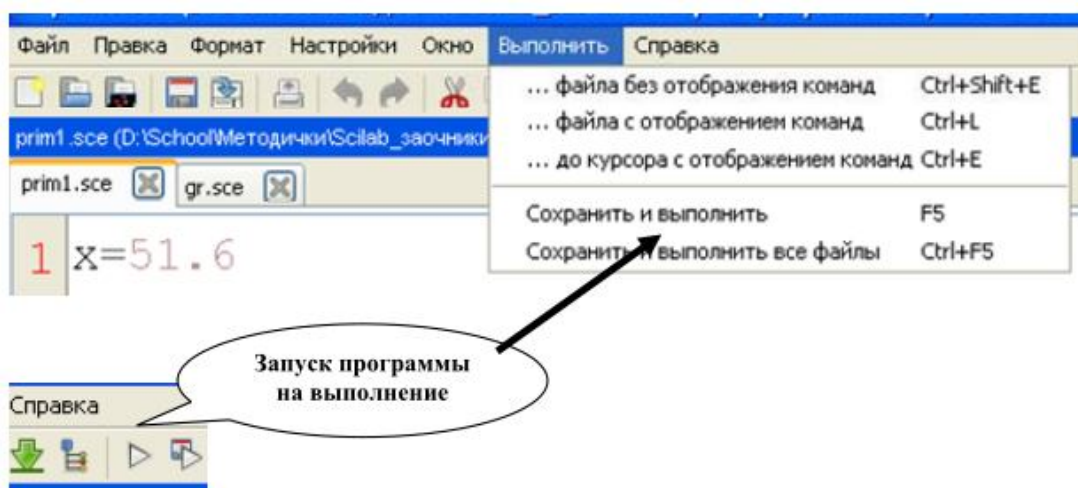


Рис.21 Запуск программы на выполнение.

Если компиляция программы прошла успешно, то результаты выполнения программы будут отражены в командном окне. Если в результате компиляции были найдены ошибки в программе, то необходимо вызвать программу в окно программы, отредактировать и повторить последовательность, начиная с шага 2.

Выражения в Scilab делятся на арифметические и логические.

Арифметическим выражением называется совокупность констант, переменных, стандартных функций, связанных знаками арифметических операций. Арифметические выражения могут содержать круглые скобки.

Ниже приведен перечень основных арифметических операций.

Название	Знак операции	Пример
Сложение	+	$x+y$
Вычитание	-	$x-y$
Умножение	*	$x*y$
Возведение в степень	^	x^5
Обратное (справа налево) деление матриц	\	$x \backslash y$
Деление матриц слева направо	/	x/y

В Scilab приоритет возведения в степень выше приоритетов умножения и деления, приоритет умножения и деления выше приоритета сложения и вычитания. Для изменения приоритета операций в математических выражениях используются круглые скобки. Степень вложения скобок не ограничивается.

Ниже приводятся некоторые часто употребляемые математические функции.

Описание	Имя	Описание	Имя
Абсолютная величина	abs(x)	Синус	sin(x)
Экспонента	exp(x)	Тангенс	tan(x)
Натуральный логарифм	log(x)	Котангенс	cot(x)
Знак числа	sign(x)	Арккосинус	acos(x)
Десятичный логарифм	log10(x)	Арксинус	asin(x)
Корень квадратный	sqrt(x)	Арктангенс	atan(x)
Косинус	cos(x)	Арккотангенс	acot(x)

Приведем примеры записи арифметических выражений в системе Scilab.

<u>Запись в математике:</u>	<u>Запись в Scilab:</u>
$\frac{\cos x^2}{x + \sin^3 x} + e^{-2.1}$	<code>cos(x^2)/(x+sin(x)^3)+exp(-2.1)</code>
$\frac{1.2 - 2x}{\lg(x + 3.2)} - \sqrt{ x - 5.8 }$	<code>(1.2-2*x)/log10(x+3.2)-sqrt(abs(x-5.8))</code>

Одним из основных операторов в Scilab является оператор присваивания. В программе этот оператор выполняет следующие функции: присваивает переменной, стоящей слева от знака «=» значение выражения, стоящего справа.

Общий вид оператора присваивания:

Имя_переменной = Выражение

В качестве параметра **Имя_переменной** может выступать имя простой переменной, структурированной переменной (вектора, матрицы), имя функции. В качестве параметра **Выражение** применяется арифметическое, логическое или строковое выражение.

Ниже приведен пример правильной записи оператора присваивания.

$$A = \cos(x) + c - d^2 * p^2 + 4.92$$

В SciLab в качестве оператора ввода используется функция **input**, которую, в силу ее значимости при программировании, принято называть оператором.

Она имеет следующий общий вид:

ИМЯ = input(Символьная константа)

Здесь ИМЯ – это имя простой переменной, Символьная константа – любой набор символов, заключенный в двойные кавычки.

Символьная константа, как правило, разъясняет смысловое назначение вводимой переменной. Например:

S=input("Задайте площадь")

A=input("Задайте значение A=")

Оператор выполняется следующим образом: в командном окне выводится набор символов, стоящий в скобках после input (символьная константа), выполнение программы приостанавливается, и компьютер переходит в режим ожидания; пользователь вводит константу, и введенная константа помещается в оперативной памяти в переменную, стоящую слева в операторе input.

При запуске на выполнение программы, содержащей оператор ввода, следует учитывать, что пока пользователь не ввел константу в ответ на запрос своей программы, оператор ввода продолжает свою работу. Система Scilab в это время блокирует выход и закрытие окна рабочего стола.

Если необходимо вывести данные на экран дисплея в определенной последовательности, применяется функция **disp**, которую принято называть оператором вывода. Оператор имеет следующий общий вид:

disp(Выражение)

Здесь **Выражение** – это арифметическое, логическое или символьное выражение, частным случаем которого являются константы или переменные любого типа.

Каждый новый оператор **disp** выполняет вывод с новой строки командного окна, например (переменным a, b, k, d уже присвоены числовые значения):

Фрагмент программы

c=a-b+k*d;

disp ("результат="), disp(c);

Командное окно

результат=

28

Выполните пример по рис.22.

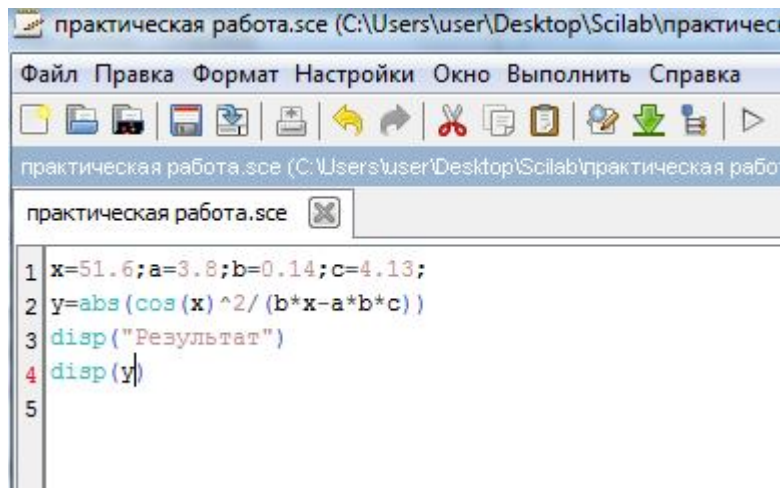


Рис.22. Пример задания.

Если действия выполнены правильно, то в командном окне вы увидите протокол выполнения программы (рис.23).

```
--> exec('C:\Users\user\Desktop\Scilab\практическая работа.sce')

Результат

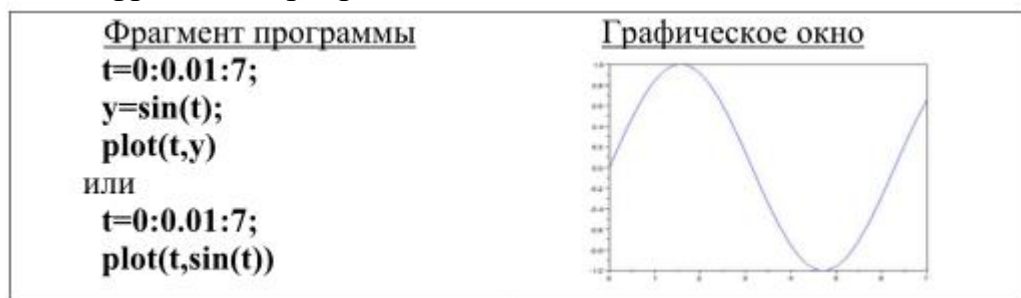
0.0109008
```

Рис.23. Результат работы программы.

Для построения графиков функций одной переменной в декартовой системе координат используются различные формы команды `plot`, которая рисует графики функций по ряду точек, соединяя их отрезками прямых.

Команда `plot(X,Y)` – строит график функции, координаты точек которой берутся из векторов одинаковой размерности X и Y . Если Y – матрица, то строится семейство графиков по данным, содержащимся в столбцах матрицы.

Например, для построения графика функции $y=\sin(t)$ нужно задать следующий фрагмент программы.



MathCAD – это популярная система компьютерной математики, предназначенная для автоматизации решения массовых математических задач в различных областях науки, техники и образования.

Для запуска формульного редактора следует установить указатель мыши в любом свободном месте окна и щелкнуть левой кнопкой, тем самым, указав

место вставки вычислительного блока. При этом курсор ввода в виде маленького крестика окажется перенесенным в это место. В процессе создания формулы курсор ввода превращается в синий уголок, указывающий направление и место ввода. Для расширения охваченной уголком области и выделения тем самым выражения следует использовать клавишу <Пробел>.

Подготовка вычислительных блоков облегчается благодаря выводу шаблона при задании того или иного оператора. Для этого в **MathCAD** служат палитры математических знаков и шаблонов операторов и функций.

Сложные математические выражения наряду с операторами содержат математические функции. **MathCAD** имеет множество встроенных элементарных, специальных и статистических функций, вставка которых обеспечивается выбором команды **Вставить функцию...** (*Function...*) из меню **Вставка** (*Insert*) либо кнопкой панели инструментов.

Помимо встроенных функций **MathCAD** предоставляет пользователю возможность для определения собственных функций согласно следующему алгоритму:

1. ввести имя функции и левую круглую скобку;
2. ввести список аргументов, отделяемых друг от друга запятыми, и закончить его правой круглой скобкой;
3. ввести двоеточие, что приведет к появлению знака присваивания $:=$ и следующего за ним поля ввода;
4. напечатать в поле ввода выражение, которое соответствует данной функции.

Все переменные, используемые в выражении, задающем функцию, должны быть определены заранее или входить в список аргументов. Для определения переменной следует:

- ввести имя переменной;
- ввести двоеточие, что приведет к появлению знака присваивания $:=$ и следующего за ним поля ввода;
- напечатать в поле ввода число или выражение, значение которого и будет присвоено переменной.

При этом следует помнить, что имена переменных могут состоять из латинских и греческих букв, цифр и символа подчеркивания. Начинаться имя переменной может только с буквы, символы нижнего и верхнего регистров различаются. Следует также отметить, что значение переменной доступно только правее и ниже её определения.

Введём для примера параболическую функцию, предварительно описав три константы a, b, c .

$$a := 1 \quad b := 1 \quad c := -1 \quad f(x) := a \cdot x^2 + b \cdot x + c$$

Теперь, для того чтобы получить значение функции, достаточно записать:

$$f(0) = -1 \quad f(1.5) = 2.75$$

Обращаем внимание на разницу в использовании оператора присваивания $:=$ и знака $=$, который задает команду **Вывести значение**.

Также следует заметить, что в качестве разделителя целой и дробной части числа используется точка.

Помимо рассмотренных выше вычислительных возможностей в **MathCAD** также предусмотрен механизм многократного повторения вычислений, (подобно циклам в языках программирования). Для этого было введено понятие интервальной переменной в формате:

:= начальное_значение[, начальное_значение + шаг].конечное значение

в скобках указан необязательный параметр шаг, по умолчанию равный 1.

Двоеточие ":" вводится клавишей точка с запятой "<;>".

Введём для примера интервал изменения аргумента x на отрезке $[-2;2]$ с шагом $h=0.1$.

$x := -2, -1.9 .. 2$

Фактически мы получаем набор из 41 значения аргумента. Чтобы убедиться в этом, достаточно ввести $x =$.

Для того чтобы вывести таблицу значений функции, следует ввести $f(x)$ и знак "=".

Например.

$a := 1 \quad b := 1 \quad c := -1 \quad f(x) := a \cdot x^2 + b \cdot x + c \quad x := -2, -1.9 .. 2$

$f(x) =$

1
0.71
0.44
0.19
-0.04
-0.25
-0.44
-0.61
-0.76
-0.89
-1
-1.09
-1.16
-1.21
-1.24
-1.25

Для построения графика функции в **MathCAD** следует воспользоваться графической палитрой, для вызова которой предназначена кнопка .

На раскрывшейся палитре выбрать кнопку  "х-у график", после чего на экране появится шаблон для построения декартова графика (рис. 24).

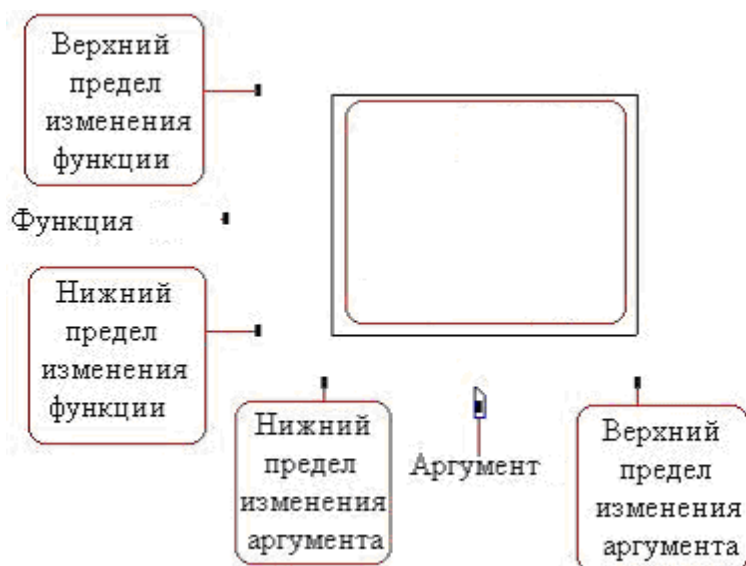


Рис. 24. Шаблон для построения декартова графика.

Далее в маркере оси x указать переменную x и функцию $f(x)$ в маркере оси y . Заканчивается построение нажатием клавиши <Enter> или щелчком мыши вне графика (см. рис.25).

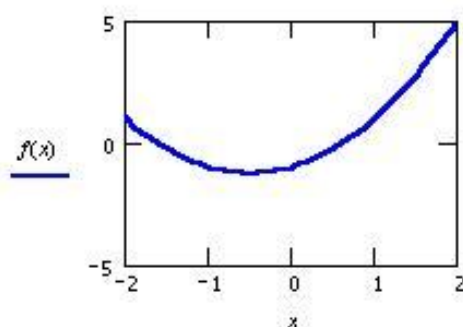


Рис. 25. Результат построения декартова графика функции.

Можно также явно указать начальное и конечное значение по осям в маркерах начала и конца оси, иначе они определятся автоматически. Выделив график двойным щелчком мыши, можно произвести его настройку, в частности, определить тип, цвет и толщину линии, а также выбрать оси.

Для решения системы линейных алгебраических уравнений (СЛАУ) в программе **MathCAD** можно применить встроенную функцию `isolve`. Для этого система уравнений должна быть записана в матричной форме $Ax=b$:
`isolve (A, b)` — вектор решения системы линейных алгебраических уравнений:

A — матрица коэффициентов системы;

b — вектор правых частей.

Пример 1. Численное решение СЛАУ.

$$A := \begin{pmatrix} 1 & 2 & -3 \\ 4 & 5 & 6 \\ 7 & -8 & -9 \end{pmatrix} \quad b := \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$$

$$\text{lsolve}(A, b) = \begin{pmatrix} 4.667 \\ 1.333 \\ -0.889 \end{pmatrix}$$

$$|A \cdot \text{lsolve}(A, b) - b| = 0$$

Пример 2. Символьное решение СЛАУ.

$$A := \begin{pmatrix} 1 & 2 & -3 \\ 4 & 5 & 6 \\ 7 & -8 & -9 \end{pmatrix} \quad b := \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$$

$$\text{lsolve}(A, b) \rightarrow \begin{pmatrix} \frac{14}{3} \\ \frac{4}{3} \\ -\frac{8}{9} \end{pmatrix}$$

2.9. Основы алгоритмизации и программирования

Алгоритм – формальное описание вычислительной процедуры. Исходные данные, передаваемые процедуре, называются входом алгоритма. Результат работы процедуры есть выход алгоритма.

Алгоритм может быть записан самыми различными способами: в виде блок-схем, текста программ и т.п.

Блок-схемы оформляются в соответствии с ГОСТ 10.002-80 ЕСПД и ГОСТ 10.003-80 ЕСПД, где операции обозначаются в виде блоков, большая часть которых вписана в прямоугольники размером $a \times b$.

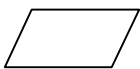
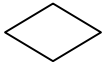
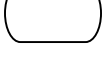
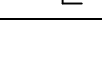
При этом минимальный размер a равен 10мм, увеличение a осуществляется с шагом 5мм, а величина b определяется как $1,5 \times a$. Для отдельных блоков допускается соотношение между a и b один к двум. При переносе блоков используется соединитель – окружность диаметром $0,5 \times a$.

Линии, соединяющие блоки, указывают на последовательность связей между ними. Стрелка в конце линии может не ставиться, если линия направлена слева направо или сверху вниз. В блок может входить несколько линий. Из блока (кроме логического) может выходить только одна линия. Все блоки нумеруются.

Виды и назначение основных блоков приведены в Таблице 2.

Основные блоки алгоритмов.

Таблица 2.

	Процесс
	Ввод-вывод
	Решение
	Предопределенный процесс
	Подготовка, модификация
	Документ
	Магнитный диск
	Пуск-останов (начало-конец)
	Соединитель
	Межстраничный соединитель
	Комментарий

Общие сведения о языке Паскаль.

В языке Паскаль используются следующие *символы*:

- буквы латинского алфавита;
- цифры от 0 до 9;
- специальные символы: :=; +; -; *; /; >; <; =; (); []; .; ; ; {}
- ключевые слова: and, array, begin, const, do, end, then, var и др.;
- знаки операций: + - / * и др.

В языке Паскаль выделяют следующие *категории лексем*:

- *специальные символы*: + - * / < = > [] () , : ; \$:= .. < = > = и др.;
- *зарезервированные (ключевые) слова*;
- *идентификаторы* – имена констант, типов, переменных, процедур, функций, программ, меток и др.;
- *метки*: числовые и символьные;

- числа*: целые десятичные, целые шестнадцатеричные и вещественные десятичные;
- строка символов*;
- комментарий*.

В качестве *разделителей лексем* применяются пробел, символ табуляции и составной символ перехода в начало следующей строки.

Общая структура программы на языке Паскаль:

Общая структура программы на языке Pascal	Пример программы
{ I. Заголовок программы }	
program Имя_Программы;	program summa;
{ II. Раздел указания используемых модулей }	
uses Список_Используемых_Модулей;	uses crt;
{ III. Раздел описаний }	
label Описание_Меток;	
const Описание_Констант;	const n=3.5;
type Описание_Типов;	
var Описание_Переменных;	var a, summa as real;
procedure Описание_Процедур;	
function Описание_Функций;	
exports Описания_Экспортируемых_имен;	
{ IV. Раздел операторов (операторный блок) }	
begin	begin
Операторы	clrscr; read(a); summa:=a+n; write('summa=', summa);
end.	end.

Стандартные типы данных в языке Паскаль:

	Название типа	Идентификатор	Диапазон значений	Размер памяти
Целые типы	Короткое целое со знаком	Shortint	-128...127	1 байт
	Целое со знаком	Integer	-32768...32767	2 байта
	Длинное целое со знаком	Longint	-2147483648 ... 147483647	4 байта

	Название типа	Идентификатор	Диапазон значений	Размер памяти
	Целое без знака	Word	0...65535	2 байта
	Короткое целое без знака	Byte	0..255	1 байт
Вещественные типы	Вещественное	Real	от $2.9 \cdot 10^{-39}$ до $1.7 \cdot 10^{38}$	6 байт
	Вещественное двойной точности	Double	от $5.0 \cdot 10^{-324}$ до $1.7 \cdot 10^{308}$	8 байт
Булевские типы	Булевское	Boolean	True или false	1 байт
Символьные типы	Символьное	Char	1 символ таблицы ASCII	1 байт
Строковые типы	Строковое	String	Последовательность символов ASCII длиной до 255	На 1 символ 1 байт

Приведем некоторые часто используемые *пользовательские типы данных* языка Паскаль:

–*перечисления* — описывают новые типы данных, значения которых определяет пользователь. Список элементов этого типа заключается в круглые скобки;

–*интервальный тип* — диапазон значений какого-то порядкового типа, называемого базовым. Список допустимых значений этого типа задается через указание минимального и максимального значений из некоторого диапазона через две точки;

–*структурированный тип* служит для описания данных, которые могут иметь много значений: тип-массив, множество, запись, файловый тип.

Описание операций.

Операция — конструкция в языках программирования, аналогичная по записи математическим операциям, то есть специальный способ записи некоторых действий. Наиболее часто применяются арифметические, логические и строковые операции.

В отличие от функций, операции часто являются базовыми элементами языка и обозначаются различными символами пунктуации, а не алфавитно-цифровыми.

Арифметические операции:

* умножение

/ деление

+ сложение

- вычитание

div — целочисленное деление

mod — остаток от целочисленного деления

Операции отношения:

= равно

<> не равно

<= меньше или равно

>= больше или равно

< меньше

> больше

Логические операции:

not логическое отрицание

and логическое умножение (и)

or логическое сложение (или)

xor логическое исключающее или

Строковая операция:

+ конкатенация (сцепление) строк

Операции инкремента и декремента:

Inc(A) — инкремент, увеличение значения A на единицу

Dec(A) — декремент, уменьшение значения A на единицу

Inc(A,n) — увеличение значения A на n

Dec(A,n) — уменьшение значения A на n

Старшинство операций:

1. приоритет: not
2. приоритет: *, /, div, mod, and
3. приоритет: +, -, or, xor
4. приоритет: =, <>, <, >, <=, >=, in

Встроенные математические функции.

abs(x)	возвращает абсолютное значение (модуль) x
sqr(x)	возвращает квадрат x
sqrt(x)	возвращает квадратный корень из x
sin(x)	возвращает синус x
cos(x)	возвращает косинус x
ln(x)	возвращает натуральный логарифм x
exp(x)	возвращает e в степени x (e=2.718281...)
arctan(x)	возвращает арктангенс x
power(x,y)	возвращает x в степени y
round(x)	возвращает результат округления x до ближайшего целого
trunc(x)	возвращает целую часть x
int(x)	возвращает целую часть x
frac(x)	возвращает дробную часть x

Правила записи выражений.

В состав арифметического выражения на языке Паскаль могут входить числовые константы, имена переменных, знаки математических операций, математические функции и функции, возвращающие число, открывающиеся и закрывающиеся круглые скобки. Правила построения выражений напоминают математические с некоторыми уточнениями.

Арифметические выражения записываются по следующим правилам:

- выражения записываются в строчку;
- все знаки арифметических операций прописываются в явном виде; нельзя опускать знак умножения между сомножителями и ставить рядом два знака операций;
- для обозначения переменных используются буквы латинского алфавита и символ подчеркивания;
- аргументы всех встроенных функций пишутся в круглых скобках;
- количество открывающихся и количество закрывающихся скобок должно быть одинаковым;
- операции выполняются в порядке старшинства: сначала вычисление функций, затем возведение в степень, потом умножение и деление и в последнюю очередь — сложение и вычитание;
- для изменения порядка операций в выражении используются круглые скобки;
- операции одного старшинства выполняются слева направо.

Операторы ввода данных.

Read(x1,x2,...,xn);

Осуществляет ввод x1,x2, ...,xn с клавиатуры.

Readln(x1,x2,...,xn);

Осуществляет ввод x1,x2, ...,xn с клавиатуры и после выбора значения последней переменной обеспечивает переход к началу новой строки.

Readln;

Обеспечивает пропуск одной строки и переход к началу новой строки.

Операторы вывода данных.

Write(x1,x2,...,xn);

Выполняет вывод значений, соответствующих перечисленным именам, размещая выводимые значения в одной строке.

Writeln(x1,x2,...,xn);

Выполняет вывод значений x1,x2,...,xn, соответствующих перечисленным именам, и после вывода последнего значения осуществляет переход к новой строке.

Writeln;

Обеспечивает пропуск строки и переход к началу новой строки.

Условный оператор.

Синтаксис условного оператора:

-короткая форма условного оператора:



if условие then оператор;

if условие then оператор1 else оператор2;

Цикл с параметром.



ИЛИ

ИЛИ

оператор;

(шаг -1)

```

graph TD
    Start((...)) --> Condition{условие}
    Condition -- да --> Body[тело цикла]
    Body --> Condition
    Condition -- нет --> Exit((...))
  
```

ИЛИ

while условие do

```

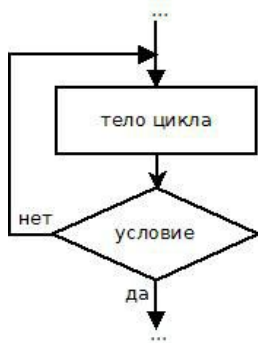
begin
    оператор1;
    оператор2;

    .... ....
    операторN;
end;

```

Если в цикле требуется
выполнять более
одного оператора

Цикл с постусловием



```

repeat
    оператор1;
    оператор2;
    .... .... ....
    операторN
until условие

```

Работа с массивами.

Массив — переменная структурированного типа. Массиву присваивается имя, посредством которого можно ссылаться на него, как на единое целое, так и на любой его элемент.

Каждое из значений, составляющих массив, называется его *элементом*.

Элементы массива располагаются в последовательных ячейках памяти, обозначаются именем массива и индексом.

Размерность массива — количество индексов элементов массива. По размерности массивы делятся на одномерные (линейные), двумерные, трёхмерные и т.д.

Размер массива — количество элементов в нём.

Переменная с индексом — идентификатор элемента массива.

Индекс может быть константой, переменной или выражением порядкового типа (целочисленный, логический, символьный, перечислимый, диапазон).

Примеры:

одномерный массив — массив А, состоящий из элементов, каждый из которых является целым числом:

1-й элемент	2-й элемент	3-й элемент	4-й элемент	5-й элемент	6-й элемент	...
12	34	98	24	1	0	...

двумерный массив — массив В, состоящий из элементов, каждый из которых является вещественным числом:

	1-й столбец	2-й столбец	3-й столбец	4-й столбец	5-й столбец	...
1-я строка	1,2	4,78	0	0,23	67	...
2-я строка	2,3	3,45	9	0,12	0	...
3-я строка	1,12	2,1	5,89	3	0,25	...
4-я строка	0	0	5,45	4	1,56	...
5-я строка	78	2	2,45	2,1	7,8	...
...	...	...	...	...	...	...

Формат объявления одномерного массива в разделе переменных:

var <имя массива>: array[<тип индекса>] of <базовый тип>;

Пример:

const n=100;

var A: array[1..n] of integer;

Формат объявления одномерного массива в разделе типов:

const n=<число>;

type <имя типа>=array[<тип индекса>] of <базовый тип>;

var <имя массива>:<имя типа>;

Пример:

const n=100;

type mas=array[1..n] of integer;

var A: mas;

**var < имя массива>: array[< тип индекса 1 >, <тип индекса 2>] of
<базовый тип>;**

Пример:

const m=30;

n=50;

var A: array[1..m, 1..n] of real;

Формат объявления двумерного массива в разделе типов:

**type <имя типа>= array[<тип индекса 1>, <тип индекса 2>] of <ба-
зовый тип>;**

var <имя массива>:<имя типа>;

Пример:

const m=30;

n=30;

type matr=array[1..m, 1..n] of real;

var A: matr;

Процедуры и функции в языке Паскаль.

Подпрограмма – это именованный набор описаний и операторов, выполняющих определенную задачу. Информация, передаваемая в подпрограмму для обработки, называется *параметрами*, а результат вычислений – *значениями*. Обращение к подпрограмме называют *вызовом*. Перед вызовом подпрограммы должны быть обязательно описана в разделе описаний. *Описание подпрограммы* состоит из заголовка и тела. В *заголовке* объявляется имя подпрограммы и в круглых скобках ее параметры, если они есть (для функции необходимо сообщить тип возвращаемого ею результата).

Тело подпрограммы следует за заголовком и состоит из описаний и исполняемых операторов. Константы, переменные, типы данных могут быть объявлены как в основной программе, так и в подпрограммах различной степени вложенности.

Переменные, константы и типы, объявленные в основной программе до определения подпрограмм, называются *глобальными*, они доступны всем функциям и процедурам. Переменные, константы и типы, описанные в какой-либо подпрограмме, доступны только в ней и называются *локальными*.

Для правильного определения области действия идентификаторов (переменных) необходимо придерживаться следующих правил:

- каждая переменная, константа или тип должны быть описаны перед использованием;
- областью действия переменной, константы или типа является та подпрограмма, в которой они описаны;
- все имена в пределах подпрограммы, в которой они объявлены, должны быть уникальными и не должны совпадать с именем самой подпрограммы;
- одноименные локальные и глобальные переменные – это разные переменные, обращение к таким переменным в подпрограмме трактуется как обращение к локальным переменным (глобальные переменные недоступны);
- при обращении к подпрограмме доступны объекты, которые объявлены в ней и до ее описания.

Формальные и фактические параметры. Передача параметров в подпрограмму.

Обмен информацией между вызываемой и вызывающей функциями осуществляется с помощью механизма передачи параметров.

Переменные, указанные в заголовке подпрограммы, называются *формальными параметрами*. Эти переменные могут использоваться внутри подпрограммы.

Список переменных в операторе вызова подпрограммы – это *фактические параметры*.

Механизм передачи параметров обеспечивает обмен данных между формальными и фактическими параметрами, что позволяет выполнять подпрограмму с различными данными. Между фактическими параметрами в

операторе вызова и формальными параметрами в заголовке подпрограммы устанавливается взаимно однозначное соответствие. *Количество, типы и порядок следования формальных и фактических параметров должны совпадать.*

Формальные параметры процедуры можно разделить на два класса: *параметры-значения* и *параметры-переменные*.

При передаче данных через *параметры-значения* в подпрограмму передаются значения фактических параметров, и доступа к самим фактическим параметрам из подпрограммы нет.

При передаче данных *параметры-переменные* заменяют формальные параметры, и, следовательно, в подпрограмме есть доступ к значениям фактических параметров.

Описание процедуры имеет вид:

```
procedure имя_процедуры(формальные_параметры);  
label  
описание_меток;  
const  
описание_констант;  
type  
описание_типов;  
var  
описание_переменных;  
begin  
//Тело процедуры.  
end;
```

Примеры заголовков процедур.

```
procedure name_1(r:real; i:integer; c:char);
```

Однотипные параметры могут быть перечислены через запятую:

```
procedure name_2(a,b:real; i,j,k:integer);
```

Список формальных параметров не обязателен и может отсутствовать:

```
procedure name_3;
```

Если в заголовке процедуры будут применяться *параметры-переменные*, то перед ними необходимо указывать служебное слово `var`, перед *параметрами-значениями* слово `var` отсутствует:

```
procedure name_4(x,y:real; var z:real);  
//x, y – параметры-значения,  
//z – параметр-переменная.
```

После заголовка идет *тело процедуры*, которое состоит из раздела описаний (константы, типы, переменные, процедуры и функции, используемые в процедуре) и операторов языка, реализующих алгоритм процедуры.

Для обращения к процедуре необходимо использовать *оператор вызова*:

имя_процедуры (список_фактических_параметров);

Фактические параметры в списке оператора вызова отделяются друг от друга запятой:

a := 5.3;

k := 2;

s := 'a';

name_1(a, k, s);

Если в описании процедуры формальные параметры отсутствовали, то и при вызове их быть не должно:

name_3;

Описание функции состоит из заголовка и тела:

function имя_функции (формальные_параметры): тип;

label

описание_меток;

const

описание_констант;

type

описание_типов;

var

описание_переменных;

begin

//Тело функции.

end;

Заголовок функции содержит служебное слово function, любой допустимый в языке Паскаль идентификатор - имя_функции;

имена формальных параметров и их типы, разделенные точкой с запятой - формальные_параметры, тип возвращаемого функцией значения - тип (функции могут возвращать скалярные значения целого, вещественного, логического, символьного или ссылочного типа).

Примеры описания функций:

function fun_1 (x:real):real;

function fun_2(a, b:integer):real;

Тело функции состоит из раздела описаний и операторов языка, реализующих ее алгоритм. В теле функции всегда должен быть хотя бы один оператор, присваивающий значение имени функции.

Например:

function fun_2(a, b:integer):real;

begin

fun_2:=(a+b)/2;

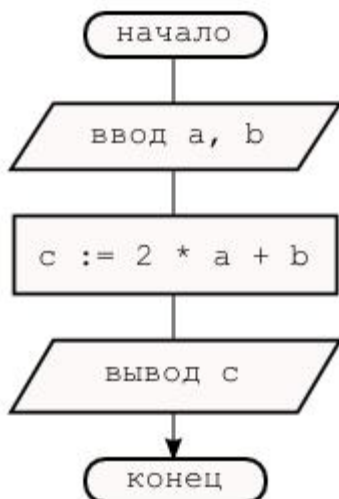
end;

Обращение к функции осуществляется по имени с указанием списка фактических параметров, разделенных запятой:
имя_функции (список_фактических_параметров);

2.10. Задача №9 (Выполнение программы линейной структуры)

Приведем пример программы линейной структуры. Вычислить значение выражения $c = 2*a + b$ и вывести на экран результат вычисления.

Блок-схема алгоритма



Программа
program line;
var a, b, c: integer;
begin
 readln (a,b);
 c := 2 * a + b;
 writeln ('c = ', c);
end.

Разделы программы:
заголовок программы
объявление переменных
начало блока операторов
ввод исходных данных
вычисление по формуле
вывод результата
конец программы (конец блока операторов).

2.11. Задача №10. (Использование циклов).

Найти приближенное решение нелинейного уравнения методом Ньютона на заданном отрезке. Точность вычислений E задается.

Рассмотрим методы решения задачи.

Метод Ньютона (касательных) является наиболее часто применяемым методом уточнения корней, пригодным для решения алгебраических и трансцендентных уравнений.

Будем считать, что искомый корень x_0 уравнения отделен отрезком $[a,b]$, на котором функция $f(x)$ удовлетворяет следующим условиям:

1. непрерывна вместе со своими производными до второго порядка включительно;
2. на концах отрезка принимает значения различных знаков;
3. производные отличны от нуля и сохраняют определенный знак.

Геометрически это означает, что график функции $y = f(x)$ в любой точке отрезка $[a, b]$ имеет касательную и не имеет экстремумов и точек перегиба.

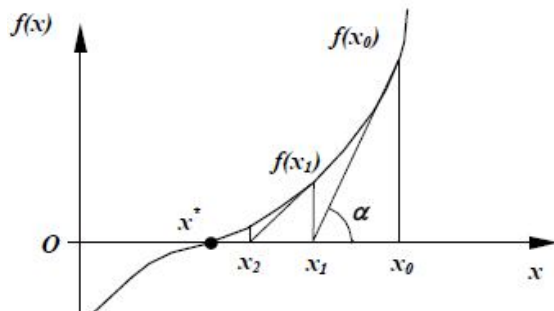


Рис.26. Метод Ньютона (касательных).

За первое приближение корня положим такое значение $x_1, x_1 \in [a, b]$, при котором знаки функции $f(x_1)$ и ее второй производной $f''(x_1)$ совпадут, и через точку $M_1(x_1, f(x_1))$ проведем касательную к кривой. Уравнение касательной

$$y - f(x_1) = f'(x_1) * (x - x_1)$$

Найдем точку x_2 пересечения касательной с осью абсцисс. Учитывая, что $y_2 = 0$, получим

$$-f(x_1) = f'(x_1) * (x_2 - x_1)$$

Отсюда получаем второе приближение корня

$$x_2 = x_1 - f(x_1) / f'(x_1)$$

Через точку $M_2(x_2, f(x_2))$ снова проводим касательную к кривой, точка пересечения которой с осью OX даст нам третье приближение корня, и т. д.

Продолжив описанный процесс построения касательных и вычисления точек их пересечения с осью OX , получим формулу итерационного метода Ньютона:

$$x_{n+1} = x_n - f(x_n) / f'(x_n), n = 1, 2, 3, \dots$$

Очередное $n+1$ приближение к корню уравнения вычисляется до тех пор, пока соблюдается условие:

$$|x_{n+1} - x_n| \geq E,$$

где E – заданная погрешность вычисления корня. Если заданная точность не достигнута после определенного числа итераций, то процесс останавливается и на экран выводится соответствующее сообщение. Блок-схема алгоритма метода Ньютона приведена на рис.27.

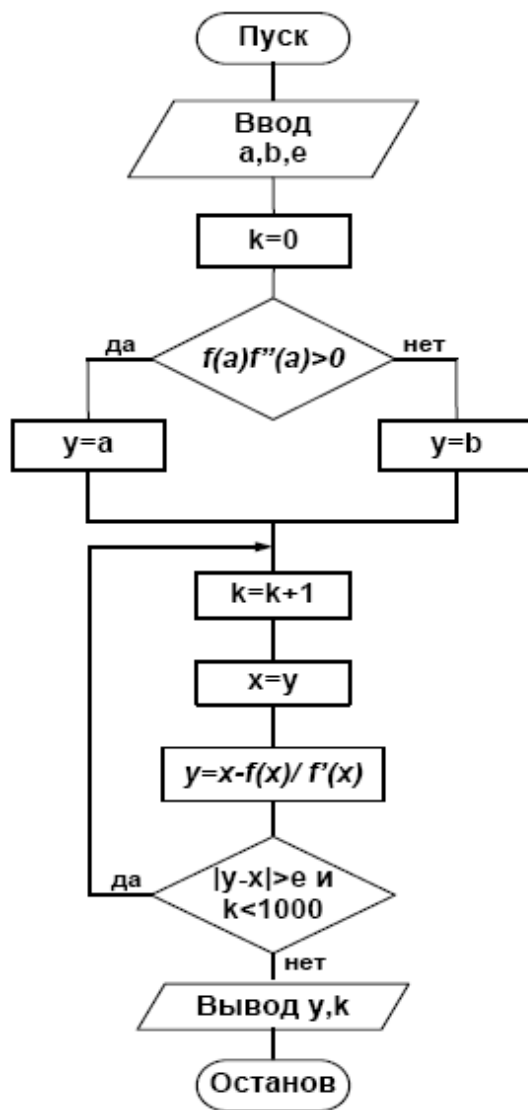


Рис.27. Блок-схема алгоритма метода Ньютона.

Метод простых итераций. Уравнение надо представить в виде $x = F(x)$. Выберем на отрезке произвольную точку x_0 - нулевое приближение к корню уравнения и примем в качестве следующего приближения

$$x_1 = F(x_0),$$

далее

$$x_2 = F(x_1),$$

n -ое приближение к корню вычисляется по формуле:

$$x_n = F(x_{n-1}).$$

Процесс последовательного вычисления чисел $x_n (n = 1, 2, \dots)$ по формуле $x_n = F(x_{n-1})$ называется методом простых итераций.

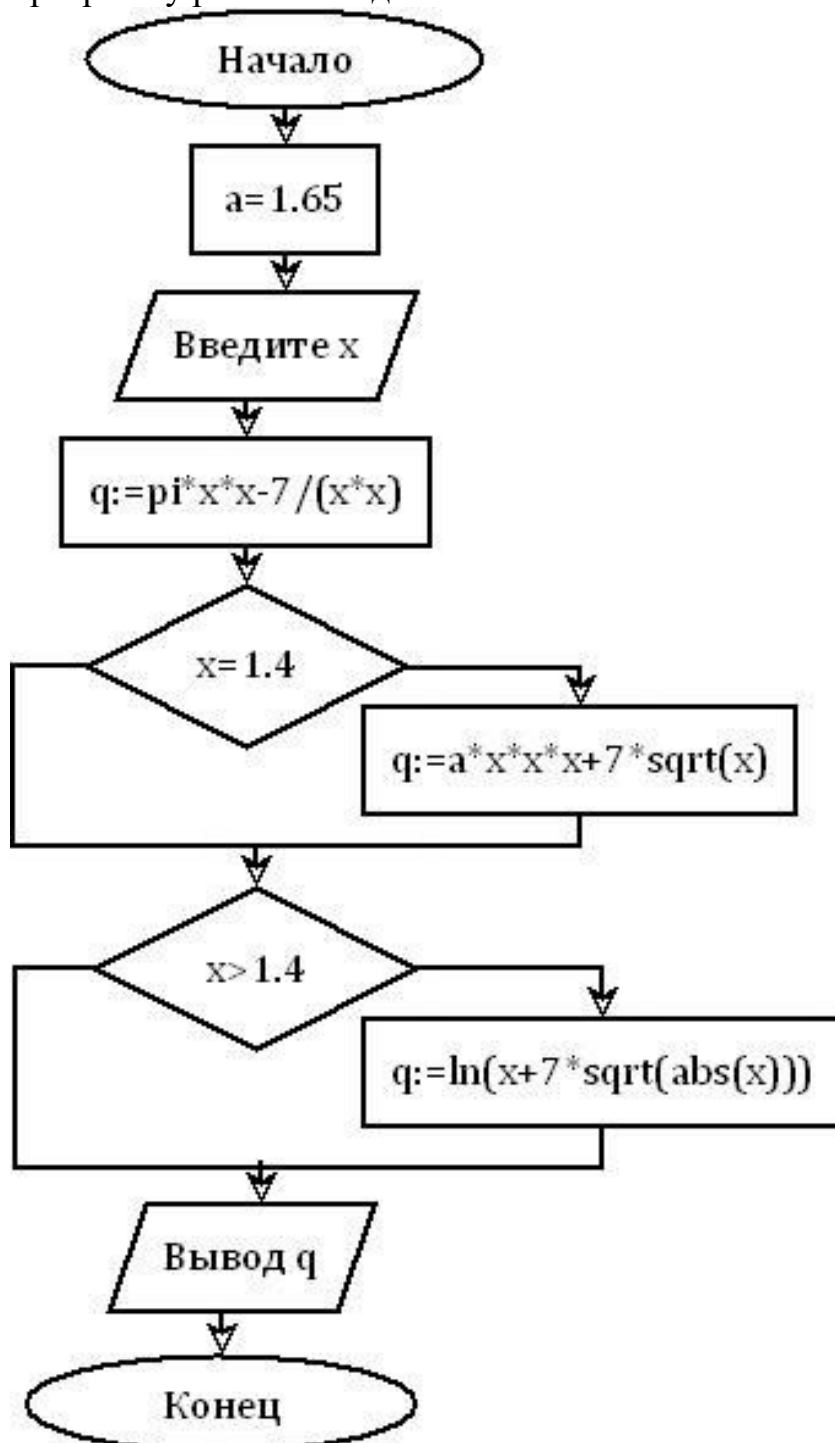
Процесс итераций следует продолжать до тех пор, пока не будет выполнено условие $|x_n - x_{n-1}| \leq E$, где E – заданная предельная абсолютная погрешность корня.

2.12. Задача №11. (Использование условного оператора)

Ввести с клавиатуры значение x , задать $a=1.65$. Вычислить значение функции:

$$q = \begin{cases} \pi * x^2 - 7 / x^2, & \text{если } x < 1.4 \\ a * x^3 + 7\sqrt{x}, & \text{если } x = 1.4 \\ \ln(x + 7 * \sqrt{|x + a|}), & \text{если } x > 1.4 \end{cases}$$

Вывести полученное значение функции на экран. Составить блок-схему и программу решения задачи.



```

program zadacha_1;
const a = 1.65;
var q, x: real;
begin
  write ('Введите x: ');
  read (x);
  q := pi * x * x - 7 / (x * x);
  if x = 1.4 then q := a * x * x * x + 7 * sqrt(x);
  if x > 1.4 then q := ln(x + 7 * sqrt(abs(x + a)));
  writeln(' q = ', q:5:2);
end.

```

Тестирование программы:

1 вариант:

Введите x: 1

q = -3.85

2 вариант:

Введите x: 1.4

q = 12.81

3 вариант:

Введите x: 2

q = 2.73

2.13. Задача №12. (Использование одномерных массивов)

Найти сумму положительных элементов одномерного вещественного массива a размера nn.

```

program zadacha_2;
const nn = 100;
var a: array[1..n] of real;
    i, n: integer;
    s: real;
begin
  writeln('Введите количество элементов массива');
  readln(n);
  writeln('Введите элементы массива');
  for i := 1 to n do
    read(a[i]);
  s := 0;
  for i := 1 to n do
    if a[i] > 0 then s := s + a[i];
  writeln('Сумма =', s);
end.

```

Пояснения к программе:

program lab6_1;	<u>Заголовок программы lab6_1</u>
const nn=100;	<u>Раздел объявления констант</u> имя nn: значение 100
var	<u>Раздел описания переменных</u>
a: array[1..nn] of real; i,n:integer; s:real;	Массив a: размер nn элементов, тип - вещественный Параметр цикла: имя i, тип – целое число Размер массива: имя n, тип – целое число Сумма положительных чисел: имя s, тип — вещественное число
begin	<u>Раздел операторов (основной блок)</u>
writeln('Введите количество элементов	Вывод на экран строки комментария 'Введите количество элементов массива'

массива');	
readln(n);	Считывание с клавиатуры целого числа и сохранение его под именем n
writeln('Введите элементы массива');	Вывод на экран строки комментария 'Введите элементы массива'
for i:=1 to n do	Организация цикла с параметром по переменной i, изменяющейся от 1 до n с шагом +1
read(a[i]);	Считывание элемента массива a с индексом i
s:=0;	Обнуление значения переменной s
for i := 1 to n do	Организация цикла с параметром по переменной i, изменяющейся от 1 до n с шагом +1
if a[i]>0 then s:=s+a[i];	Тело цикла: если элемент массива a с индексом i больше нуля, то увеличение суммы положительных чисел на значение элемента a[i]
writeln('Сумма =', s);	Вывод на экран строки «Сумма =» и значения, хранимого под именем s
end.	<u>Завершение раздела операторов (текста программы)</u>

Тестирование программы.

Введите количество элементов массива

5

Введите элементы массива

-8 7 5 0 -1

Сумма =12

2.14. Задача №13.(Использование двумерных массивов)

Найти количество положительных элементов двумерного массива x размера 5×5 .

```
program zadacha_3;
var i, j, kol: integer;
    x:array[1..5,1..5] of integer;
begin
    randomize;
    for i := 1 to 5 do
    for j := 1 to 5 do
    x[i,j] := random(20) - 10;
    for i := 1 to 5 do
    begin
        for j := 1 to 5 do
            write(x[i,j]:6);
        writeln
    end;
    writeln('*****');
    kol := 0;
    for i := 1 to 5 do
    for j := 1 to 5 do
        if x[i,j] > 0 then inc(kol);
    writeln('Количество положительных=', kol);
    end.
```

Тестирование программы.

-7 9 6 -8 -10

4 -10 0 -8 2

8 -6 3 3 -2

-7 -7 2 -1 -1

-2 -5 -5 -3 2

Количество положительных=9

2.15. Задача №14. (Использование функций)

Дана непрерывная на отрезке $[a;b]$ функция $f(x)$. Используя метод трапеций приближенного вычисления интеграла, найти $\int_a^b f(x)dx$, шаг интегрирования $h=0,01$. Для вычисления подынтегральной функции $f(x)$ написать пользовательскую функцию.

$f(x) = \sqrt{4+x^2}$ $x \in [0;3]$. Результат вычисления вывести на экран.

Метод трапеций для вычисления определенного интеграла.
 Величина определенного интеграла численно равна площади фигуры, образованной графиком функции, осью абсцисс и прямыми $x=a$, $x=b$ (геометрический смысл определенного интеграла). Следовательно, найти

$$\int_a^b f(x)dx$$

– это значит оценить площадь фигуры, ограниченной перпендикулярами, восстановленными к графику подынтегральной функции $f(x)$ из точек a и b , расположенных на оси аргумента x .

Для решения задачи разобьем интервал $[a,b]$ на n одинаковых участков. Длина каждого участка будет равна $h=(b-a)/n$.

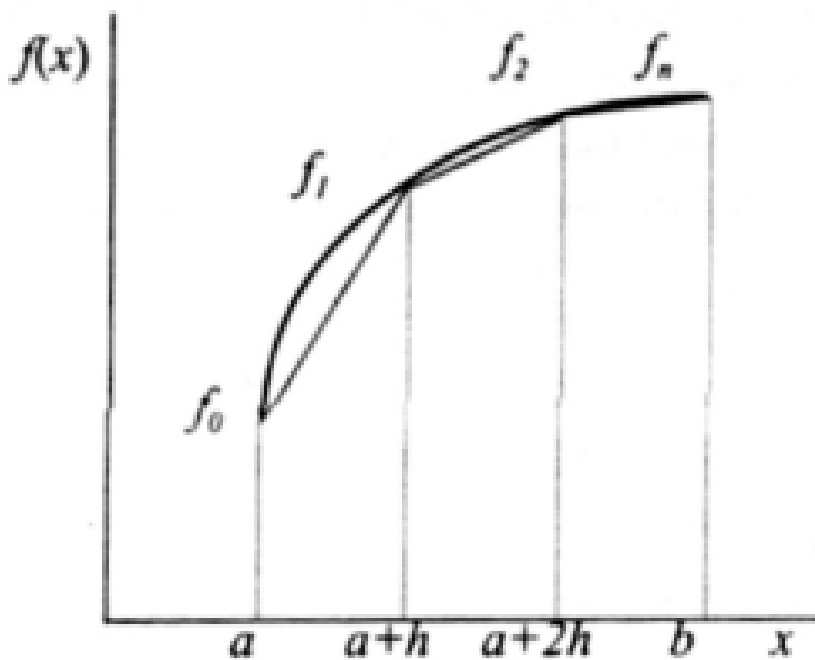


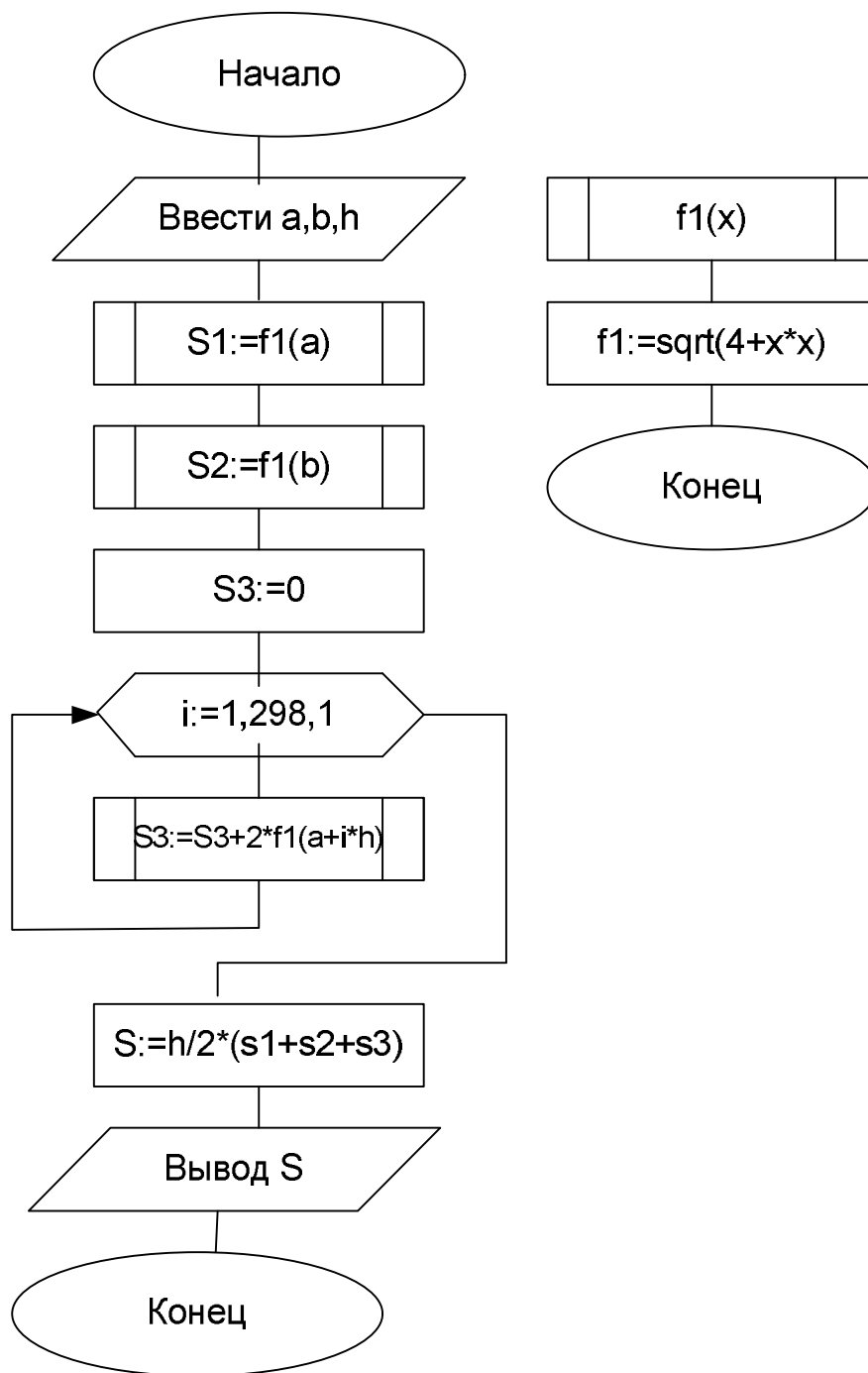
Рис.28. Геометрический смысл определенного интеграла

Формула метода трапеций

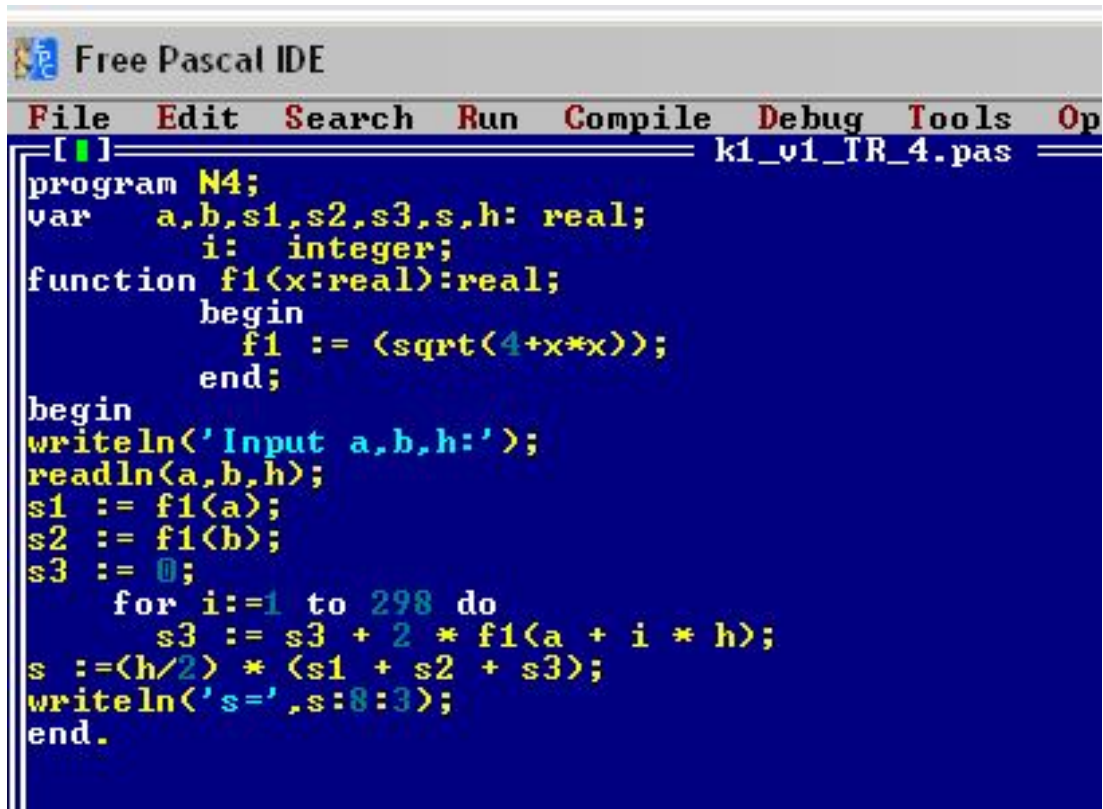
$$S = \frac{b-a}{2n} (f_0 + 2f_1 + \dots + 2f_{n-1} + f_n)$$

Найдем число частей разбиения $n = (3 - 0) / 0,01 = 300$

Блок-схема алгоритма

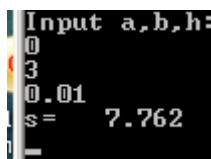


Текст рабочей программы в среде программирования Free Pascal:



```
Free Pascal IDE
File Edit Search Run Compile Debug Tools Op
k1_v1_TR_4.pas
program N4;
var  a,b,s1,s2,s3,s,h: real;
     i: integer;
function f1(x:real):real;
begin
    f1 := (sqrt(4+x*x));
end;
begin
writeln('Input a,b,h:');
readln(a,b,h);
s1 := f1(a);
s2 := f1(b);
s3 := 0;
    for i:=1 to 298 do
        s3 := s3 + 2 * f1(a + i * h);
s := (h/2) * (s1 + s2 + s3);
writeln('s=',s:8:3);
end.
```

Тестирование программы



```
Input a,b,h:
0
3
0.01
s= 7.762
```

ЛИТЕРАТУРА

1. Иopa Н.И. Информатика (для технических направлений): учебное пособие/Н. И. Иopa. – 2-е изд., стер. – М.: Кнорус, 2012. – 472 с. – (Бакалавриат).
2. Симонович С.В. Информатика. Базовый курс: Учебник для вузов. 3-е изд. Стандарт третьего поколения. – СПб.: Питер, 2015. – 640 с. : ил.
3. Акулов, О. А. Информатика: базовый курс : учеб. для вузов / О. А. Акулов, Н. В. Медведев. - 5-е изд., испр. и доп. - Москва : Омега-Л, 2008. - 574 с.
4. Захарова И.Г. Информационные технологии в образовании: учеб. пособие для вузов / И.Г.Захарова. - 6-е изд., стер. - М.: Издательский центр "Академия", 2010. - 192 с.
5. Могилев, А. В. Информатика : учеб. пособие для вузов / А. В. Могилев, Н. И. Пак, Е. К. Хеннер ; под ред. Е. К. Хеннера. - 3-е изд., перераб. и доп. - Москва : Академия, 2004. - 840 с. - (Высшее образование).
6. Панюкова С.В. Использование информационных и коммуникационных технологий в образовании: учеб. пособие для вузов / С.В.Панюкова. - М.: Издательский центр "Академия", 2010. - 224 с.
7. Фаронов, Валерий Васильевич. Турбо Паскаль: учебное пособие для студ. вузов, обуч. по направлению подгот. "Информатика и вычислительная техника" / В.В. Фаронов .— СПб. [и др.] : Питер, 2007 .— 366 с. : ил. — (Учебное пособие).

Учебное пособие

Иванова Татьяна Витальевна

ИНФОРМАТИКА

Учебное пособие

**для выполнения контрольных работ, домашних заданий
студентам 1,2 курсов всех форм обучения**

Компьютерный набор Иванова Т. В.

Усл. печ. листов 3,2