



Российский государственный университет нефти и газа
имени И.М. Губкина

Кафедра «Информатики»

Лекция 1

курс
Информатика

Рейтинговая система оценки знаний

Семестр:

- 8 лабораторных работ
- 3 контрольные работы
- 1 домашняя работа
- тесты

60

Экзамен:

- практическая часть –
(макс. 10 баллов)
- теоретическая часть –
(макс. 30 баллов)

40

Рейтинговая система оценки знаний

+ бонусные задания

(При условии выполнения учебного плана)

Организация работы

Перед лекцией: самостоятельная проработка материалов лекции

На лекции : объяснение, примеры, вопросы, обсуждение

В компьютерном классе : защита лабораторных и домашних работ, контрольные мероприятия, бонусы

Консультации

понедельник 17⁵⁰ - 19³⁰

Подготовиться!

Изучить конспект лекций и методические материалы

Организация работы

Для защиты лабораторных работ обязательно иметь тетрадь

В тетради: оформление лабораторной работы согласно заданию,
Любые конспекты и справочные материалы только в **рукописном** виде

Использование тетради: при защите лаб. работы, на тестах, контрольных и экзамене!

Организация работы

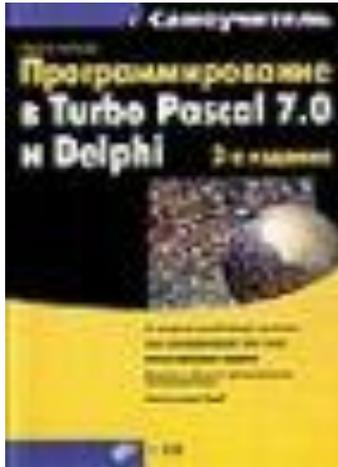
Отдельные темы предназначены для самостоятельного изучения.

Защита практической части домашних заданий проводится на консультациях в установленные сроки (доп. баллы).

Защита теоретической части домашних заданий проводится в виде тестирования на консультациях в установленные сроки + 2 недели

Организация работы

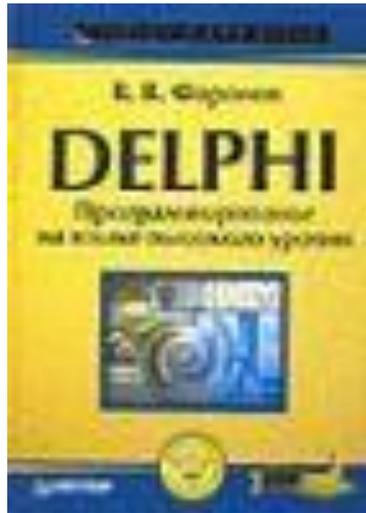
В случае пропуска контрольных мероприятий по уважительной причине студент допускается к сдаче в течение 2 недель при наличии документов.
(медицинская справка заверяется в медпункте)



Никина Культин

Программирование в Turbo Pascal
7.0 и Delphi

(2008)



Валерий Васильевич Фаронов

Delphi. Программирование на
языке высокого уровня

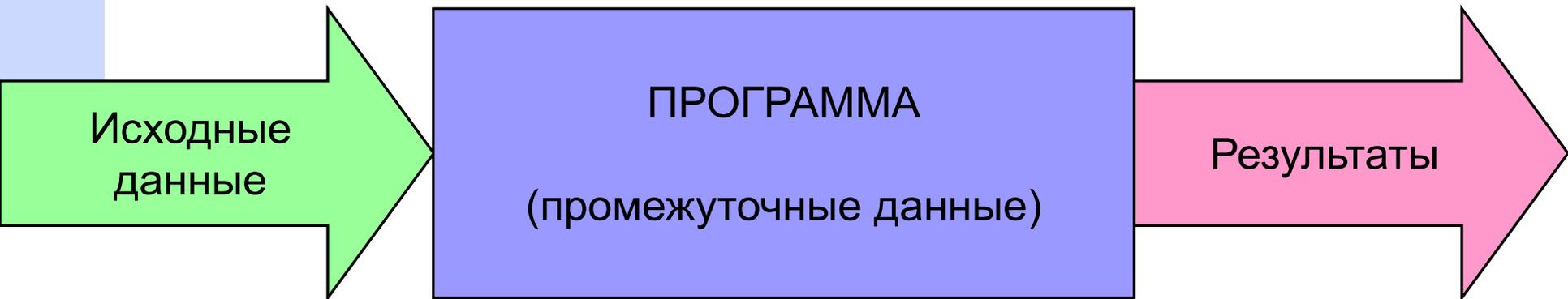
2008

Этапы решения задачи

- **Постановка задачи.**
- **Формализация задачи.**
- **Анализ объекта**
- **Разработка алгоритма.**
- **Составление программы.**
- **Отладка и тестирование программы.**
- **Расчет и анализ результатов.**
- **Возможная модификация программы.**

Постановка задачи

- Дается формулировка:
«**Что дано и что необходимо определить**».
- Четко устанавливается полный список исходных данных.
- Затем определяются требования к результатам



Анализ объекта

Результат анализа объекта – выявление его составляющих (элементарных объектов) и определения связей между ними.

Разработка информационной модели объекта.

В процессе построения модели выделяются главные, наиболее существенные, свойства, которые соответствуют цели.

Формализация

Замена реального объекта или процесса его формальным описанием, т.е. его информационной моделью.

Постановка задачи

**Масса Земли равна $6 \cdot 10^{24}$ кг,
масса Луны равна $7,3 \cdot 10^{22}$ кг,
расстояние между их центрами 384 000 км.
Определите силу тяготения между Землей
и Луной.**

Математическая модель

Закон всемирного
тяготения

$$F = G \frac{m_1 \cdot m_2}{r^2}$$

Гравитационная
постоянная

$$G = 6,67 \cdot 10^{-11} \frac{H \cdot m^2}{кг^2}$$

Формализация

■ Входные данные:

G - гравитационная постоянная $\left(\frac{\text{Н} \cdot \text{м}^2}{\text{кг}^2} \right)$

m1, m2 - массы материальных точек (кг)

r - расстояние между ними (км)

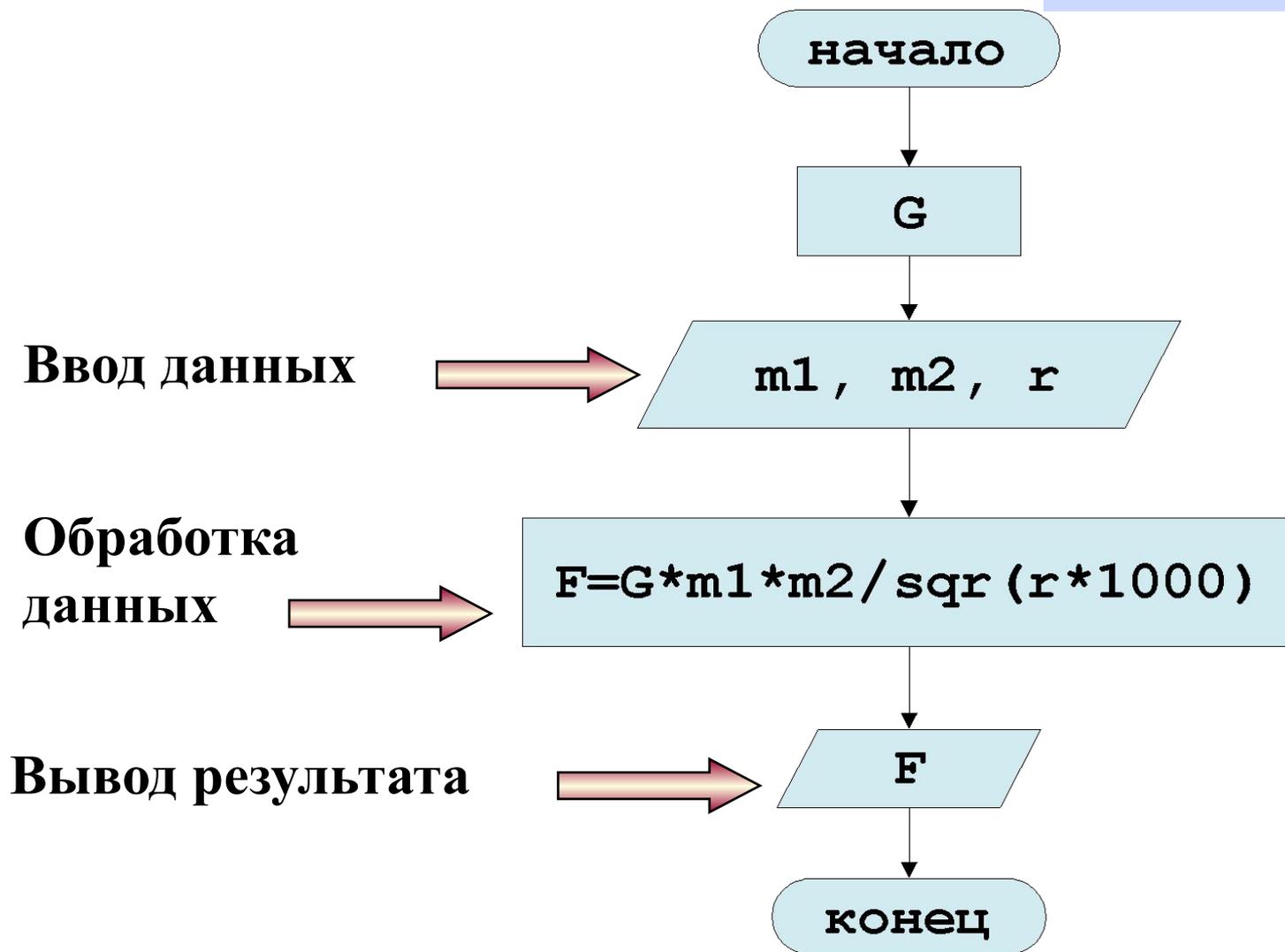
■ Обработка данных:

$$F = G * m1 * m2 / \text{sqr}(r * 1000)$$

■ Выходные данные:

F - сила тяготения (Н)

Алгоритмизация (блок-схема)



См. Методические материалы

Разработчик ПО должен знать:

- как ввести информацию в память (**ввод**);
- как хранить информацию в памяти (**данные**);
- как указать правильные команды для обработки данных (**операции**);
- как передать данные обратно из программы пользователю (**вывод**).

Среда Delphi.

- **Delphi –это среда быстрой разработки приложений (Rapid Application Development, RAD)**
- **«Быстрая» разработка базируется на технологии визуального проектирования и событийного программирования.**

Процесс создания программы в Delphi состоит из двух шагов:

1. сначала нужно создать **форму** программы (диалоговое окно),
2. затем — написать процедуры обработки **событий**.

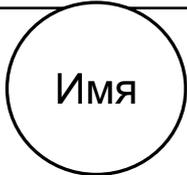
Форма приложения создается путем добавления на форму **компонентов** и последующей их настройки.

Алгоритм

- **постоянное и точное предписание (указание) исполнителю совершить определенную последовательность действий, направленных на достижение указанной цели или решение поставленной задачи.**

Свойства алгоритмов:

1. Разделение выполнения решения задачи на отдельные операции - важное свойство алгоритмов, называемое **дискретностью**.
2. Для решения задачи по заданному алгоритму, необходимо, понять и выполнить каждое действие, предписываемое командами алгоритма. Такое свойство алгоритмов называется **определенностью (или точностью) алгоритма**.
3. **Универсальность**. Алгоритм должен быть составлен так, чтобы им мог воспользоваться любой исполнитель для решения аналогичной задачи.
4. **Результативность (или конечность) алгоритма** означает, что исполнение алгоритма должно закончиться за конечное число шагов.

Название блока	Обозначение	Назначение блока
Терминатор		Начало, завершение программы или подпрограммы
Процесс		Обработка данных, (вычисления)
Данные		Операции ввода/вывода
Решение		Ветвления, выбор, итерационные и поисковые циклы
Подготовка		Счетные циклы
Предопределенный процесс		Вызов процедур
Соединитель		Маркировка разрывов линий ²³

Язык программирования Object Pascal.

Программа – набор инструкций (операторов), записанных на алгоритмическом языке и реализующих заданный алгоритм.

Элементарными синтаксическими единицами языка являются слова, которые состоят из символов алфавита, все слова отделены друг от друга специальными разделителями.

Элементы языка.

В языке Object Pascal выделены следующие основные типы слов:

- **Ключевые выражения (*зарезервированные слова*);**
- **Идентификаторы (*имена*);**
- **Значения (*литералы*).**

Алфавит языка Object Pascal

- Прописные (A-Z) и строчные (a-z) латинские буквы вместе с символом подчеркивания (_);
- цифры от 0 до 9
- ряд специальных символов.

Смысл всех литер языка в идентификаторах и служебных словах не зависит от регистра букв (прописные/ строчные).

Разделители

Разделители, не являясь символами языка, становятся значимыми, если они входят в состав строк. Разделители используются для отделения друг от друга идентификаторов, чисел, служебных слов.

В качестве разделителя могут употребляться пробелы (32) и все управляющие символы кода ASCII (0 ÷ 31).

Комментарии

В Object Pascal комментарий может быть задан в одной из трех форм:

- ❑ текст между открывающей и закрывающей фигурными скобками { и } ;
- ❑ любой текст между открывающими и закрывающими символами языка (* и *) ;
- ❑ любой текст между парой литер // и концом текущей строки.

В тексте комментариев можно использовать любые литеры, в том числе буквы кириллицы.

Идентификаторы

Идентификатор – это имя любого объекта программы (переменной, константы, процедуры, функции, типа, самой программы и т. д.).

- ✓ Имя – это последовательность латинских букв от А до Z, цифр от 0 до 9 и символов подчеркивания '_', (**не должно начинаться с цифры!**)
- ✓ строчные и прописные буквы не различаются (MAX, Max, max и т.д. одно и то же имя);
- ✓ длина идентификатора может быть произвольной, но значащими являются только первые 255 символов.

Зарезервированные слова

- ❑ заголовки: `program, unit, procedure, function;`
- ❑ блоки описания: `const, var, label, type;`
- ❑ создания НОВЫХ ТИПОВ: `array, string, record ... end, file, file of ... ;`
- ❑ операторные скобки: `begin ... end;`
- ❑ операторы
- ❑ и др.

Специальные символы

■ знаки пунктуации

{ } (* *) [] () ' / . , ^ @ # \$

■ знаки операций

+ - = * < > := >= <= <> *(комбинации спецсимволов являются единичными символами, их нельзя разделять пробелами)*

Константы.

Константы – это имена для фиксированных значений, на которые вы часто ссылаетесь в программе.

Значение констант не может измениться в процессе выполнения программы.

Константы объявляются в описательной части программы.

Раздел описания констант

const

<ИМЯ КОНСТАНТЫ>[: <ТИП>] = <значение>;

Const

```
a=17; b=-12.56e-2; c=$FF;
```

```
ch0='A'; ch1=#65;
```

```
MyName='Петр';
```

```
i: Integer=10;
```

Переменные

Переменная имеет такие характеристики:

- Имя
- Тип
- Значение
- Размещение в памяти

Прежде, чем переменные будут использоваться в программе, их необходимо объявить в разделе описания **Var**.

При объявлении переменной компилятор выделяет соответствующий объем памяти для её размещения.

Раздел описания переменных

var <список имен переменных>: <тип>;

Например:

Var

i ,k: Integer;

x ,y: Real;

S : string;

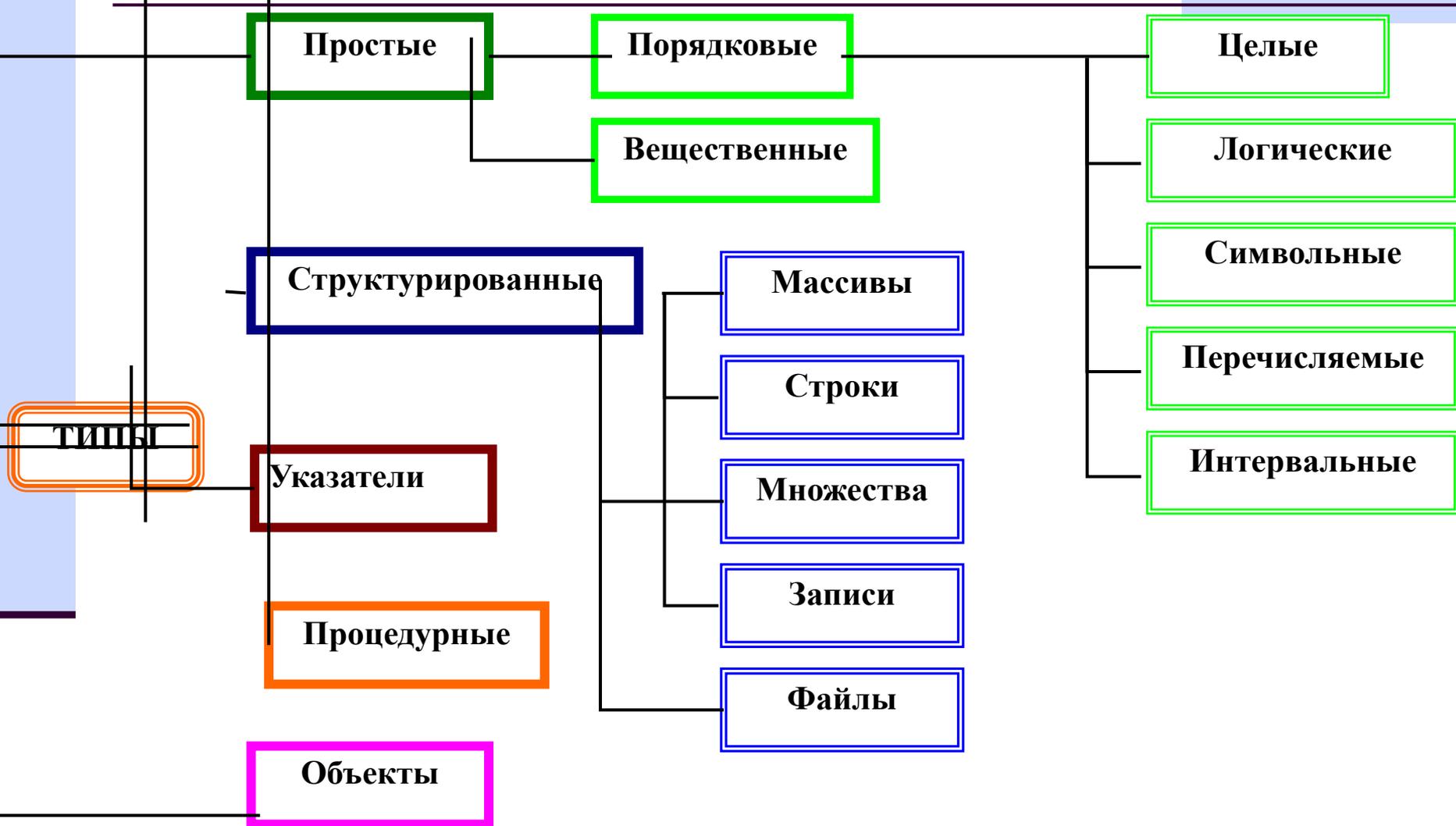
Ch : Char;

Типы данных.

С типом величины связаны три ее свойства:

- **форма внутреннего представления;**
- **множество принимаемых значений;**
- **множество допустимых операций.**

Структура типов данных



Стандартные типы:

- целый (`integer`);
- вещественный (`real`);
- логический (`boolean`);
- символьный (`char`);
- тип-строка (`string`);
- текстовый файл (`textfile`).

Простые типы данных

- **Порядковые**
- **Вещественные.**

Простые типы определяют элементы данных, состоящие только из одного компонента.

Идентификатор	Длина	Диапазон значений
	байт	
Целые типы		
Integer	2	-32768 ... 32767
Byte	1	0 ... 255
Word	2	0 ... 65535
Shortint	1	-128 ... 127
Longint	4	-2147483648 2147483647 ...

Идентификатор	Длина байт	Диапазон значений
Вещественные типы		
Real	6	$2,9 \cdot 10^{-39} \dots 1,7 \cdot 10^{38}$
Single	4	$1,5 \cdot 10^{-45} \dots 3,4 \cdot 10^{38}$
Double	8	$5 \cdot 10^{-324} \dots 1,7 \cdot 10^{308}$
Extended	10	$3,4 \cdot 10^{-4932} \dots 1,1 \cdot 10^{4932}$
Логический тип		
Boolean	1	True, false
Символьный тип		
Char	1	Все символы кода ASCII

Арифметические операции.

Тип операции	Оператор	Операция	Тип операндов	Тип результата	Пример
бинарный	+	Сложение	Integer, Real	Integer, Real	$X + Y$
	-	Вычитание			Result - 1
	*	Умножение			$Pi * Radius$
	/	Деление			$X / 2$
	div	Целочисленное деление	Integer	Integer	$13 \text{ div } 4 = 3$
	mod	Остаток от деления			$13 \text{ mod } 4 = 1$
унарный	+	Положительный знак	Integer, Real	Integer, Real	$+ 7$
	-	Отрицательный знак			$-X$ <small>42</small>

ОПЕРАТОРЫ

- ПРИСВАИВАНИЯ

Имя := Выражение

- УПРАВЛЕНИЯ: РАЗВЕТВЛЕНИЯ

If, Case

ЦИКЛЫ

For, While, Repeat

~~ПЕРЕХОДЫ~~

~~Goto~~

- ОБРАЩЕНИЯ К ФУНКЦИЯМ, ПРОЦЕДУРАМ

Оператор присваивания.

<имя переменной> := <выражение>;

Например:

x := 5;

y := 10;

x := x + y;

Выражения.

Выражение — это конструкция, которая возвращает величину.

Операция — это определенное действие над элементами данных. Сами элементы данных, над которыми выполняется операция, называются **операндами**.

Арифметическое выражение

$$b = \frac{|x - y| \cdot \left(1 + \frac{\sin^2(x)}{x + y}\right)}{e^{|x - y|} + \frac{x}{2}}$$

```
b:=abs(x-y)*(1+sqr(sin(x))/(x+y))/(exp(abs(x-y))+x/2);
```

Стандартные функции см. в
МЕТОДИЧЕСКИХ
МАТЕРИАЛАХ

Правила записи выражений.

- Все составные части выражения записываются в одну строку;
- Используются только круглые скобки;
- Два знака арифметических операций не должны записываться подряд ($n^*(-5)$).
- Несколько записанных подряд операций одинакового приоритета выполняются последовательно слева направо.
- Часть выражения, заключенная в скобки, вычисляется в первую очередь.

Целочисленное деление (DIV)

Пример 1:

найдем результат операции **11 DIV 5**

1. для этого сначала разделим 11 на 5:

$$\frac{11}{5} = 2\frac{1}{5}$$

2. отбрасываем дробную часть
3. получаем результат: **11 DIV 5 = 2.**

Целочисленное деление и остаток от деления
см. в МЕТОДИЧЕСКИХ МАТЕРИАЛАХ

Остаток от деления нацело (MOD)

$$x \bmod y = x - (x \operatorname{div} y) * y$$

Пример 1:

найдем результат операции **11 MOD 5**

1. для этого сначала получим результат операции: **11 DIV 5 = 2**
2. произведем вычисления по указанному алгоритму:
здесь **$x=11$ $y=5$ $(x \operatorname{div} y)=2$**
 $x \bmod y = 11 - 2 * 5 = 1$
3. получаем результат: **11 MOD 5 = 1.**

Структурное программирование:

- ✓ методы разработки и записи программы, которые ориентированы на максимальные удобства для восприятия и понимания человеком.
- ✓ при прочтении программы должна четко прослеживаться **логика** её работы.
- ✓ отдельные фрагменты программы представляют собой логические (управляющие) структуры, которые определяют порядок выполнения содержащихся в них правил обработки данных.

Основные логические структуры:

Следование-последовательность операторов (групп операторов), выполняемых друг за другом в порядке следования в тексте программы;

Ветвление- управляющая структура, которая в зависимости от выполнения заданного условия определяет выбор для исполнения одного из двух или более заданных в этой структуре групп операторов;

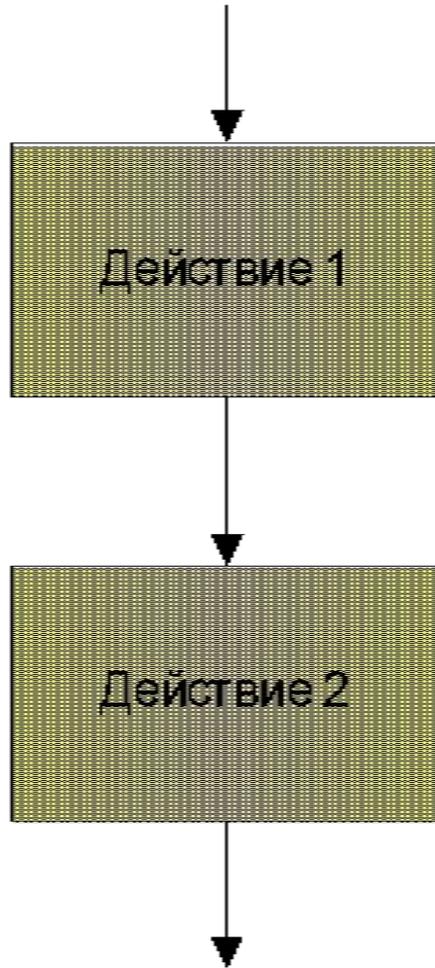
Повторение- цикл, в котором группа операторов может выполняться повторно, если соблюдается заданное условие.

Существенная особенность всех этих структур – то, что каждая из них имеет ***только один вход*** и ***только один выход***, что и обеспечивает логически последовательную структуру программы.

Все эти структуры определяются ***рекурсивно***, т. е. каждая из входящих в структуру групп операторов может быть любой из возможных структур – допускается вложение структур.

Базовые структуры алгоритмов

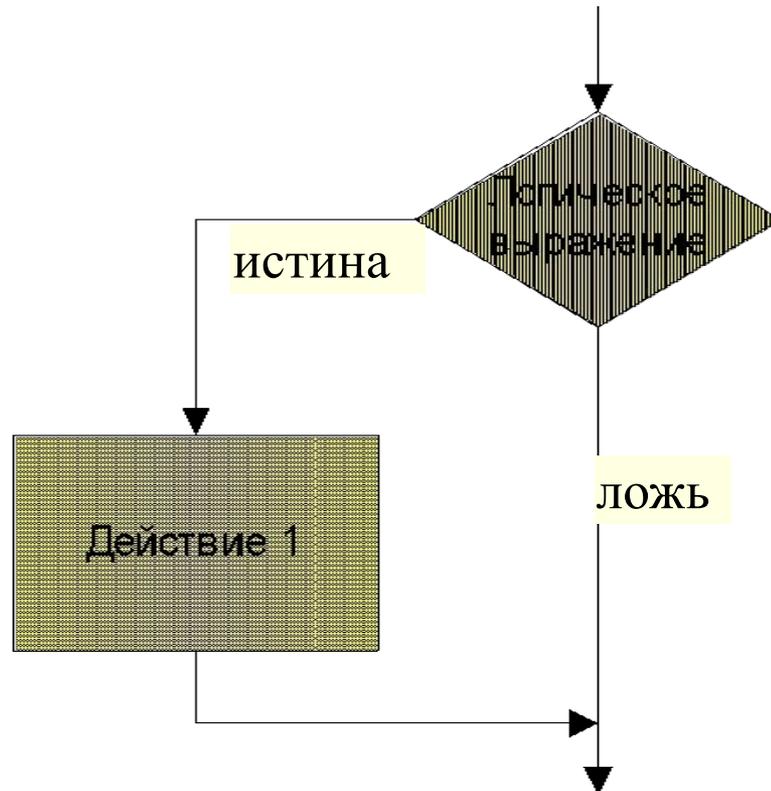
- **следование** – обозначает последовательное выполнение действий;



Структура ветвление существует в четырех основных вариантах:

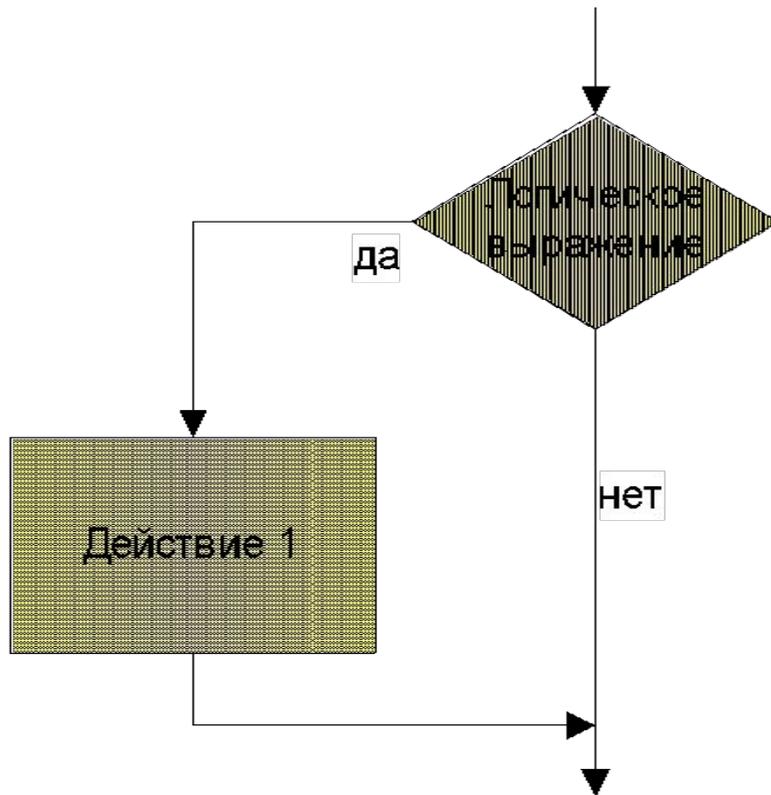
Язык блок-схем

1. ЕСЛИ - ТО



Структура ветвление существует в четырех основных вариантах:

1. если - то

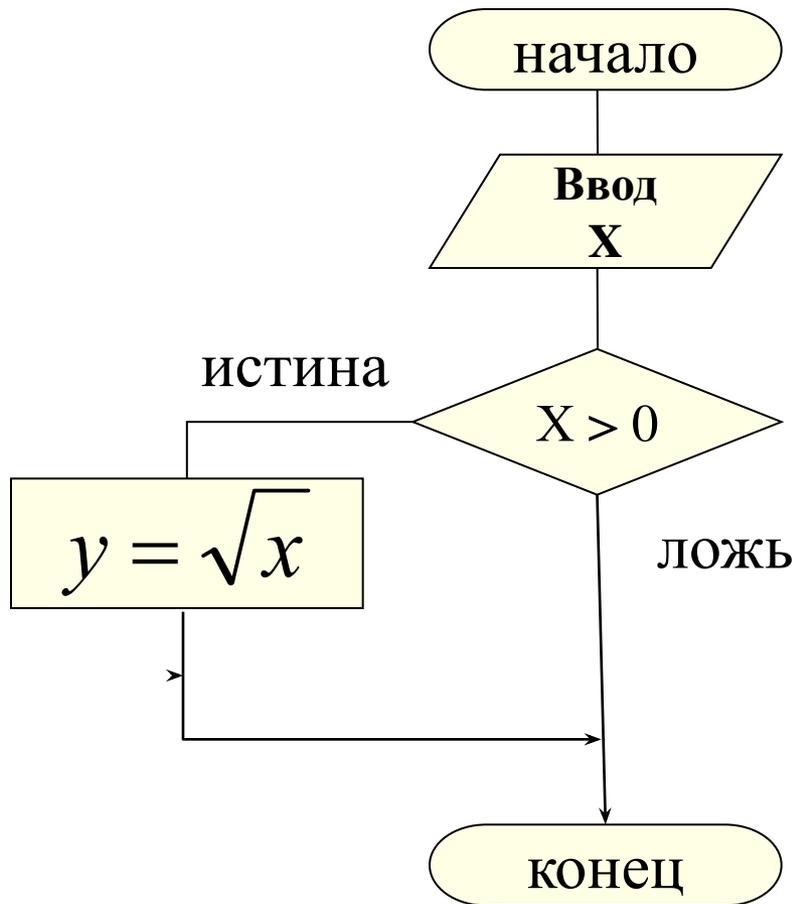


Выходя из дома, смотрим в окно: « Дождь идет ?»

Если да, то берем зонт и выходим.

Если нет, то просто выходим.

$$y = \sqrt{x}, \text{ если } x > 0$$

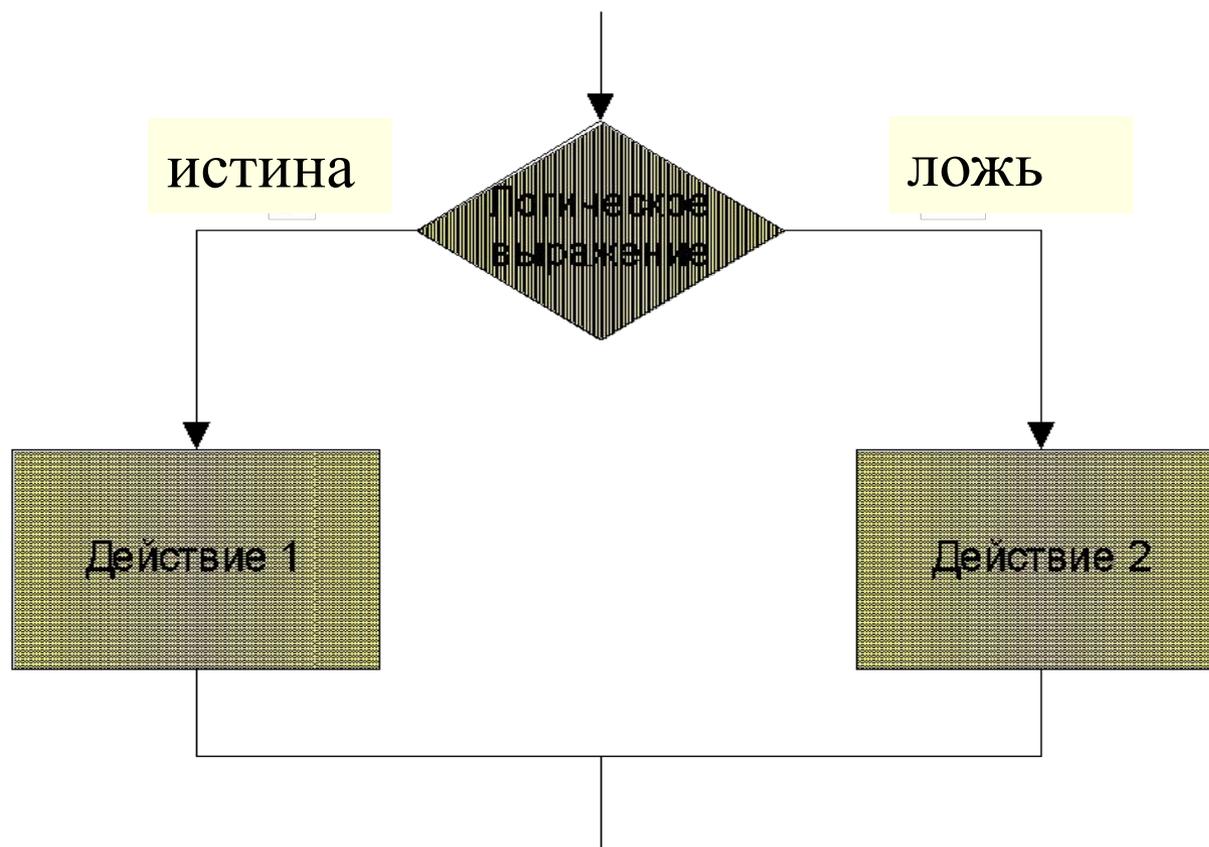


Язык DELPHI

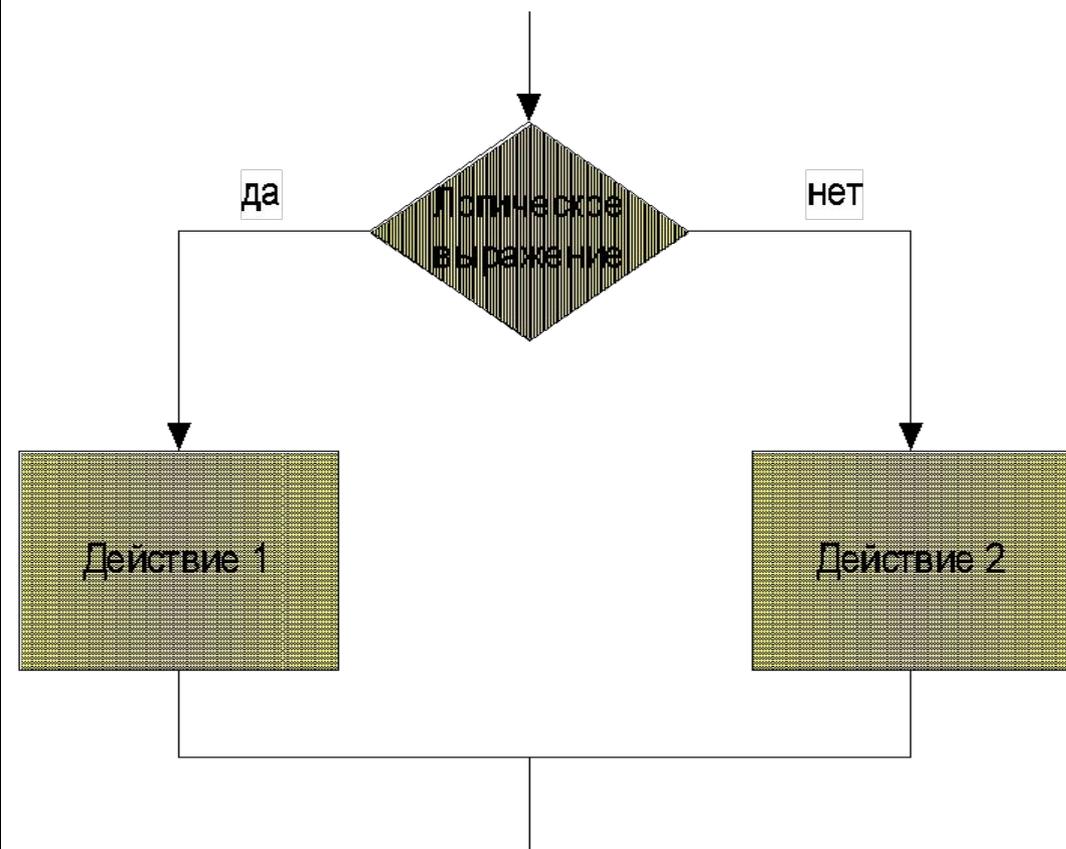
IF условие **THEN** действие 1 ;

```
If x>0 then y:=sqrt(x) ;
```

2. если - то - иначе



2. если - то - иначе

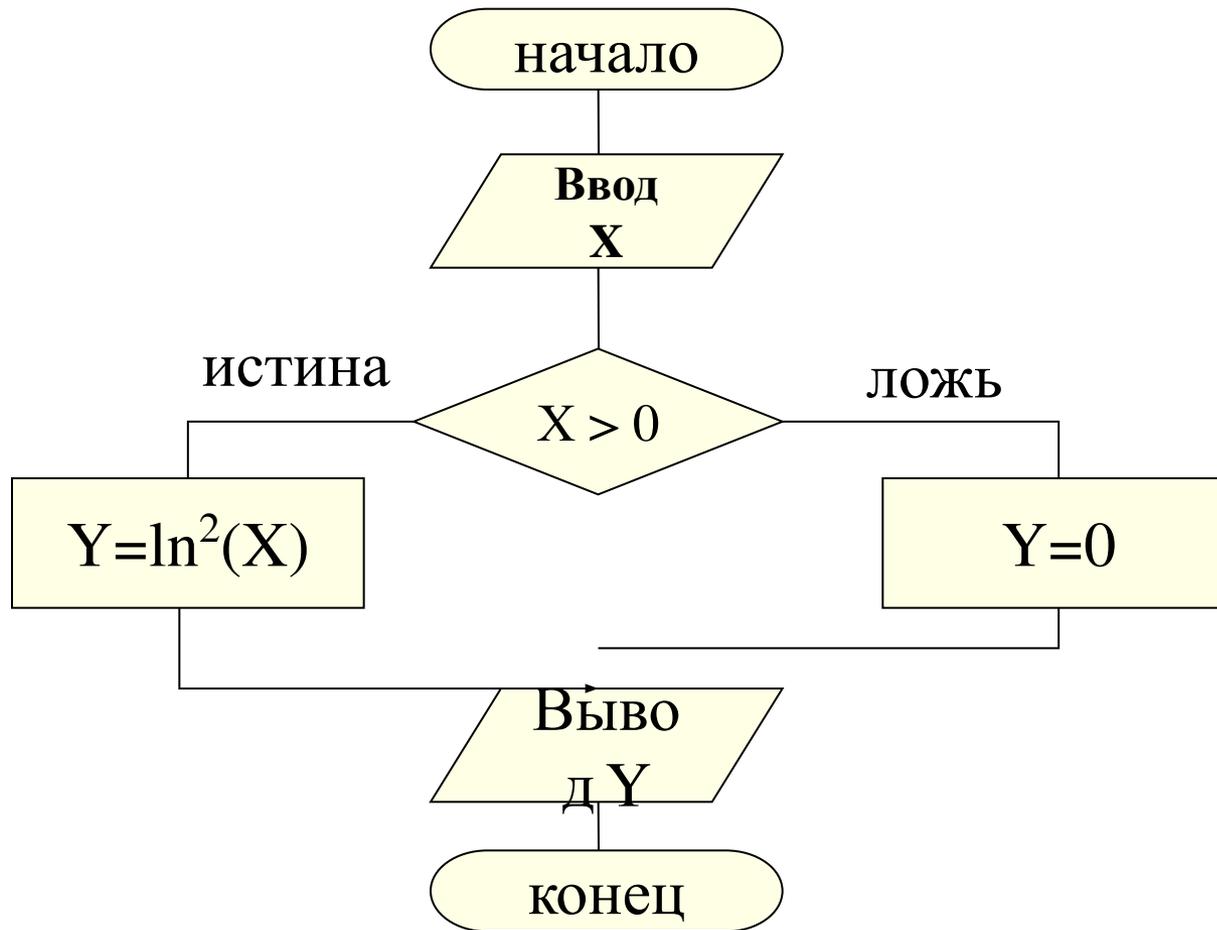


Пусть D дискриминант квадратного уравнения.

Тогда, если $D < 0$, то решение отсутствует.

В противном случае у данного квадратного уравнения решение существует

$$y = \begin{cases} 0, & \text{если } x \leq 0 \\ \ln^2 x, & \text{если } x > 0 \end{cases}$$



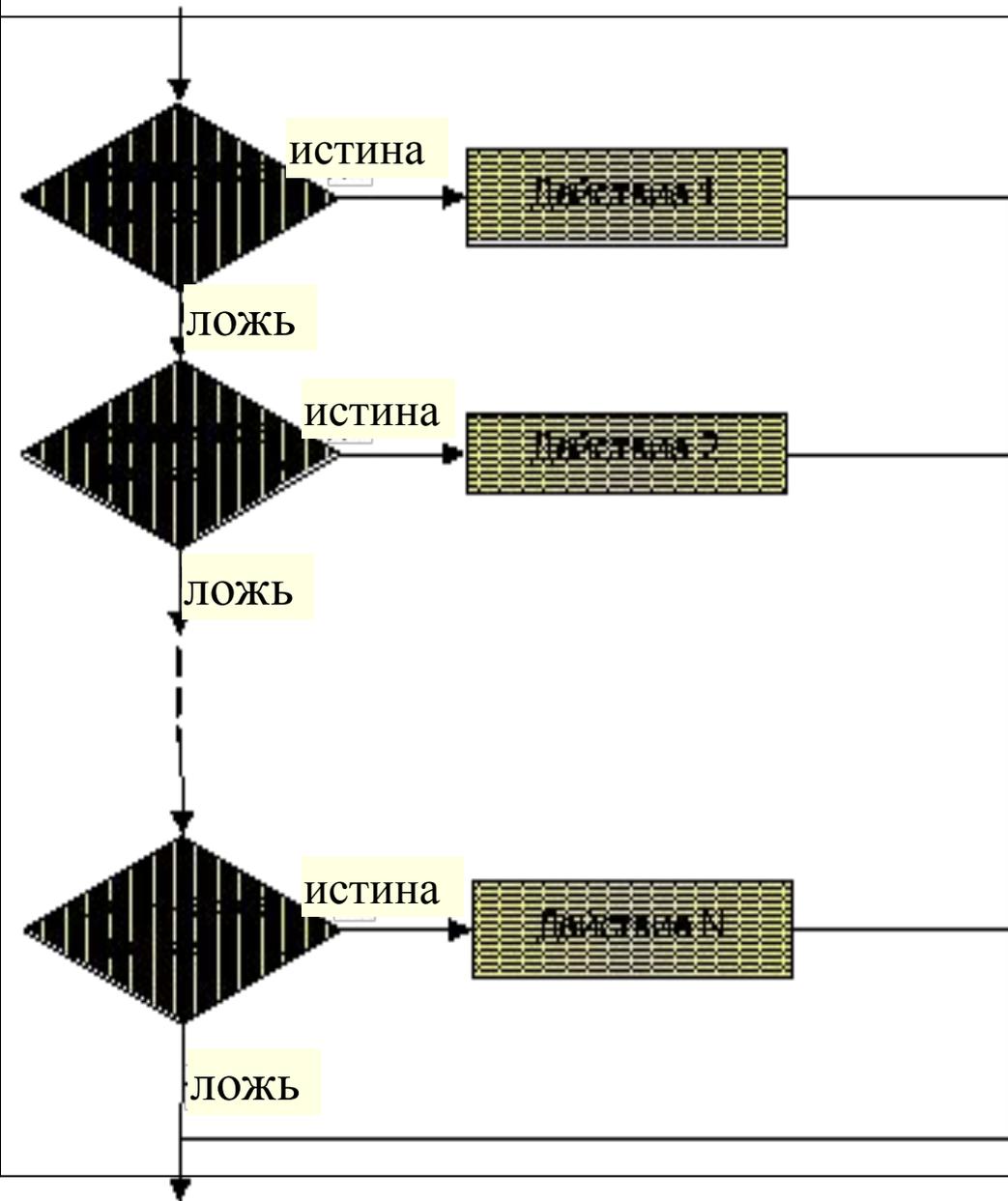
Язык DELPHI

IF условие **THEN** действие 1

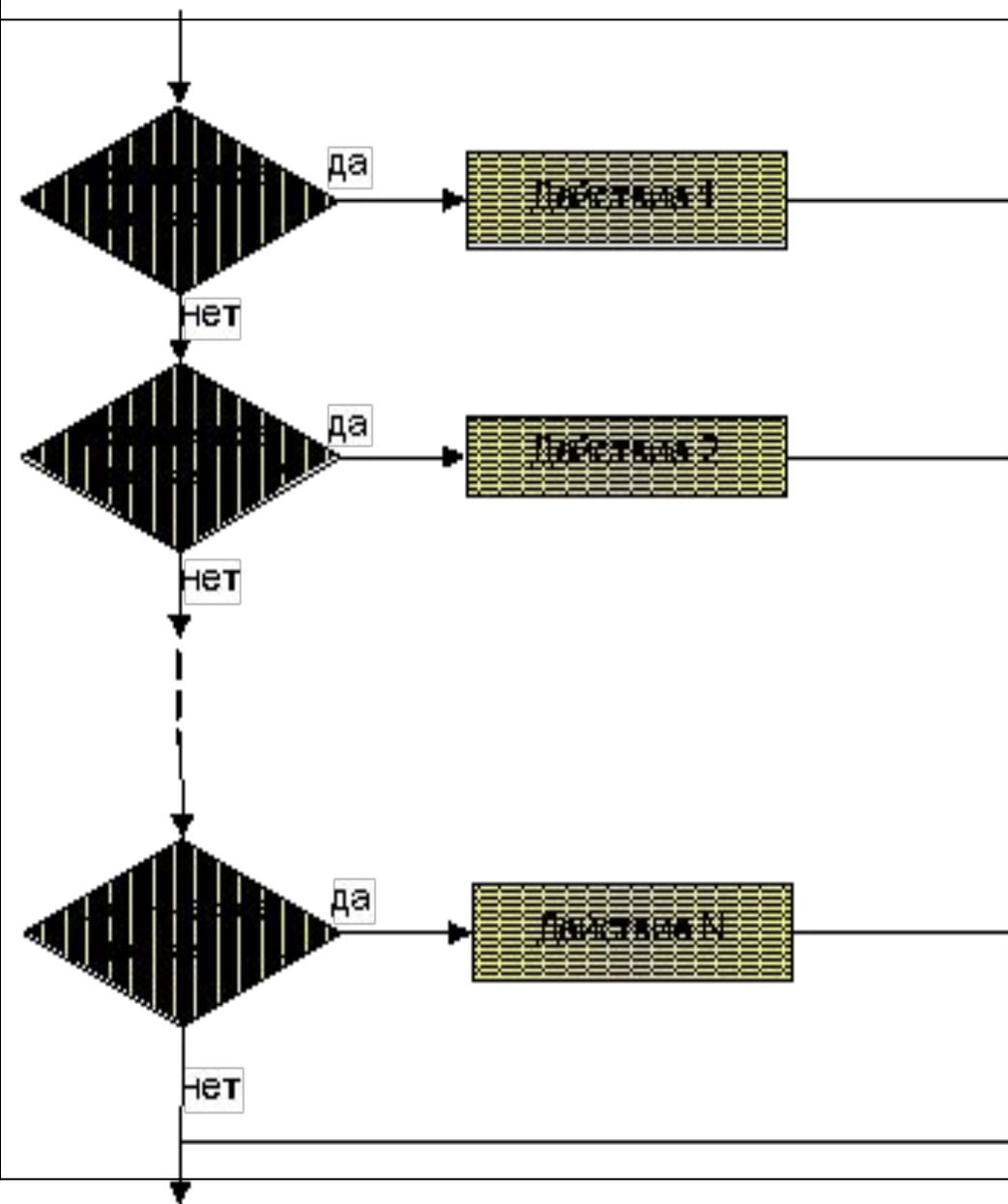
ELSE действие 2;

```
If x>0 then y:=sqr(ln(x))  
      else y:=0;
```

3. выбор



3. выбор



Вручение медалей:

1 место – золотая

2 место – серебряная

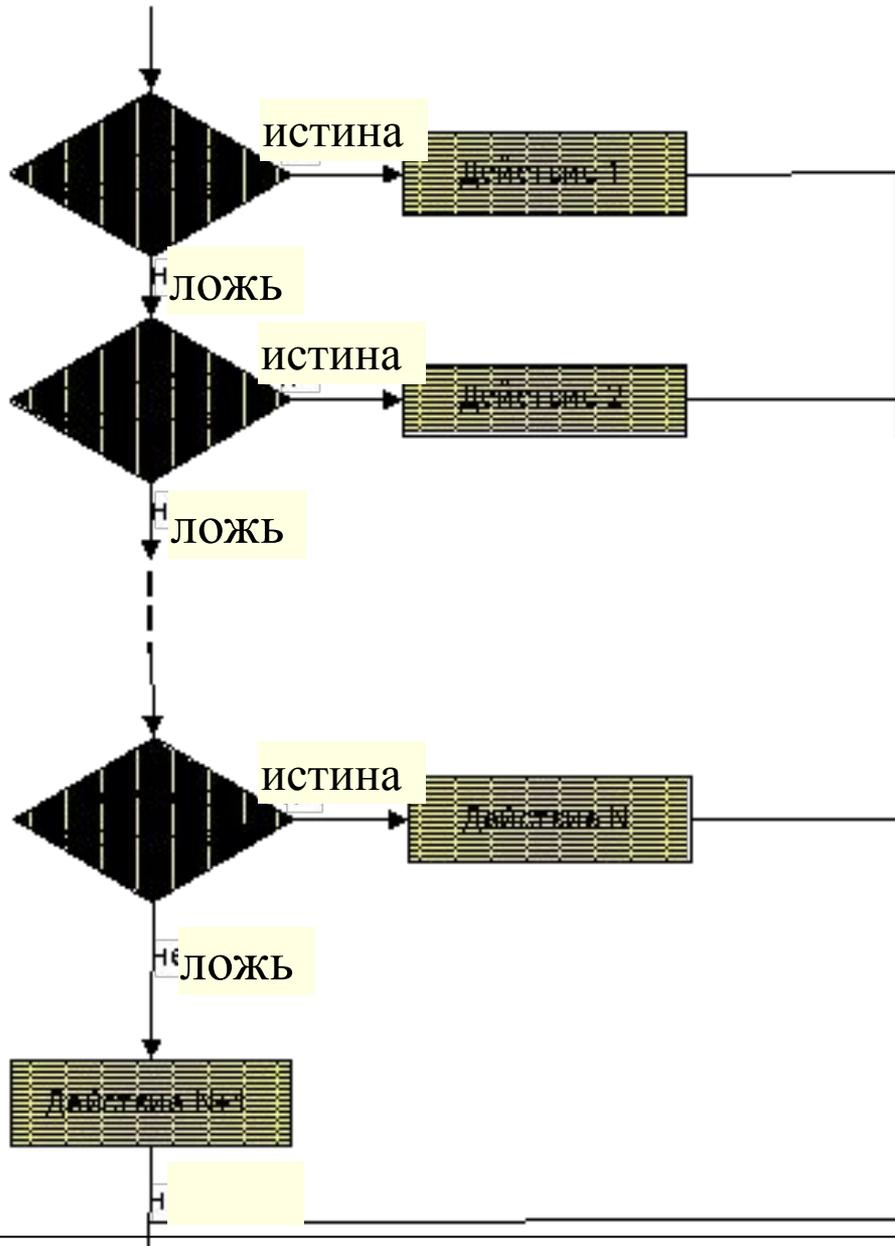
3 место - бронзовая

3. Язык DELPHI

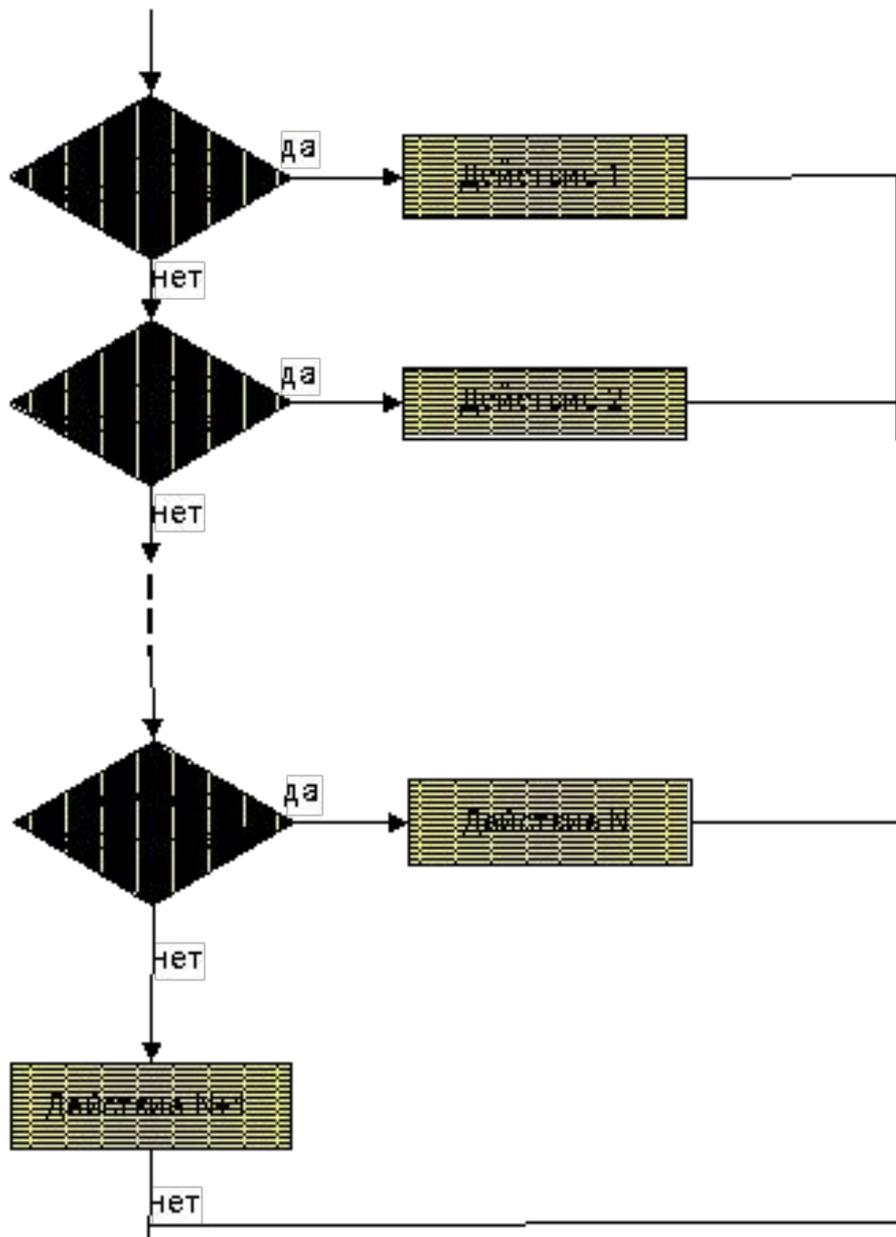
```
CASE <ключ_выбора> OF  
<константа выбора 1> :  
    действие 1;  
<константа выбора 2> :  
    действие 2;  
...  
<константа выбора N> :  
    действие  
N;  
END;
```

```
y:=3;  
k:=2;  
Case k of  
1:   y:=5;  
2:   y:=y-3;  
3:   y:=2*k;  
end;
```

4. выбор - иначе



4. выбор - иначе



**Сдача экзамена:
5 – отлично**

4 – хорошо

3 – удовлетворительно

Иначе – экзамен не сдан

3. Язык DELPHI

```
CASE <ключ_выбора> OF  
<константа выбора 1> :  
действие 1;  
<константа выбора 2> :  
действие 2;  
. . .  
<константа выбора N> :  
действие N;  
ELSE действие N+1 ;  
  
END ;
```

```
y:=3;  
k:=4;  
Case k of  
1:   y:=5;  
2:   y:=y-3;  
3:   y:=2*k;  
else y:=k+y;  
end;
```