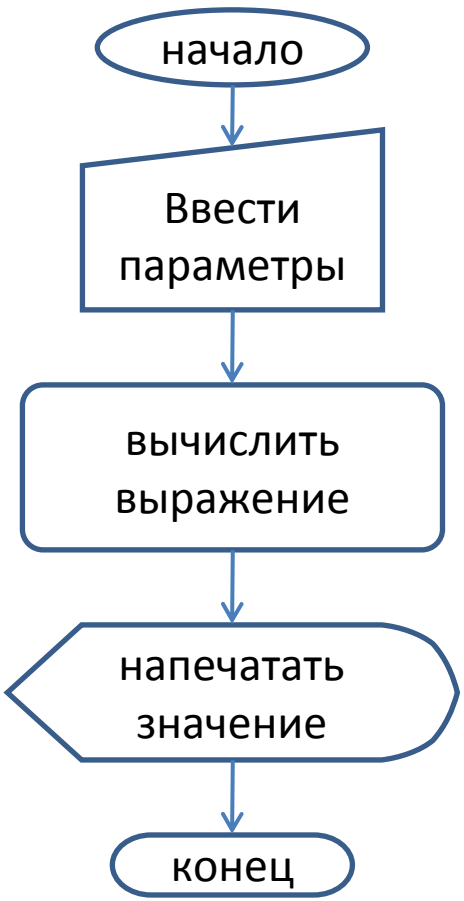
Алгоритмы и алгоритмические языки

Лекция 4

23.09.2024

Задача.
Напечатать значение выражения
если a и b – целые числа,
а c может быть вещёственным

$$\frac{a^2 - b}{c + 3{,}14159}$$



После ввода и до использования при вычислении введённые значения должны как-то храниться. Для этого используются переменные.

Задача.
Напечатать значение выражения
если a и b – целые числа,
а c может быть вещёственным

$$\frac{a^2 - b}{c + 3{,}14159}$$

Решение на языке Паскаль

```
program EXAMPLE (input,output);
var a,b: integer; c: real;
begin
    read(a,b,c);
    write( (a*a-b)/(c+3.14159) )
end.
```

Задача.
Напечатать значение выражения
если a и b – целые числа,
а c может быть вещёственным

$$\frac{a^2 - b}{c + 3{,}14159}$$

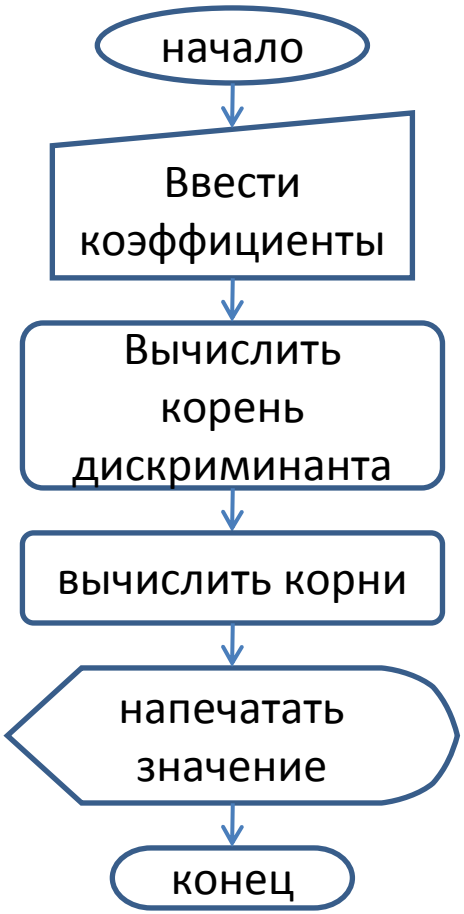
Решение на языке Си

```
#include <stdio.h>
int a;
int main()
{
    int b;
    float c;
    scanf("%d%d%f", &a,&b,&c);
    printf("%f ", (a*a-b)/(c+3.14159) );
}
```

Задача.

Напечатать значения корней квадратного уравнения $ax^2+bx+c=0$, если известно, что они определены и различны.

$$x_{1,2} = \frac{\pm\sqrt{b^2-4ac}-b}{2a}$$



$$d = \sqrt{b^2-4ac}$$

$$x_1 = \frac{d-b}{2a}$$

$$x_2 = \frac{-d-b}{2a}$$

Задача.

Напечатать значения корней квадратного уравнения $ax^2+bx+c=0$, если известно, что они определены и различны.

$$x_{1,2} = \frac{\pm\sqrt{b^2-4ac}-b}{2a}$$

Решение на языке Паскаль

$$d = \sqrt{b^2-4ac}$$

```
program sqvRoots(input, output);
var
   a, b, c, d: real;
begin
   read( a, b, c );
   d:=sqrt( sqr(b)  - 4*a*c );
   writeln('x1=',  (-b+d)/(2*a), '  x2=', (-b-d)/(2*a)  )
end.
```

$$x_{1,2} = \frac{\pm d - b}{2a}$$

Задача.

Напечатать значения корней квадратного уравнения $ax^2+bx+c=0$, если известно, что они определены и различны.

$$x_{1,2} = \frac{\pm\sqrt{b^2-4ac}-b}{2a}$$

Решение на языке Си

$$d = \sqrt{b^2-4ac}$$

$$x_{1,2} = \frac{\pm d - b}{2a}$$

```
#include <stdio.h>
#include <math.h>
int main()
{
    float a,b,c,d ;
    scanf(“%f%f%f", &a,&b,&c);
 d=sqrt( b*b - 4*a*c );
 printf("x1=%f x2=%f /n“, (-b+d)/(2*a), (-b-d)/(2*a) )
}
```

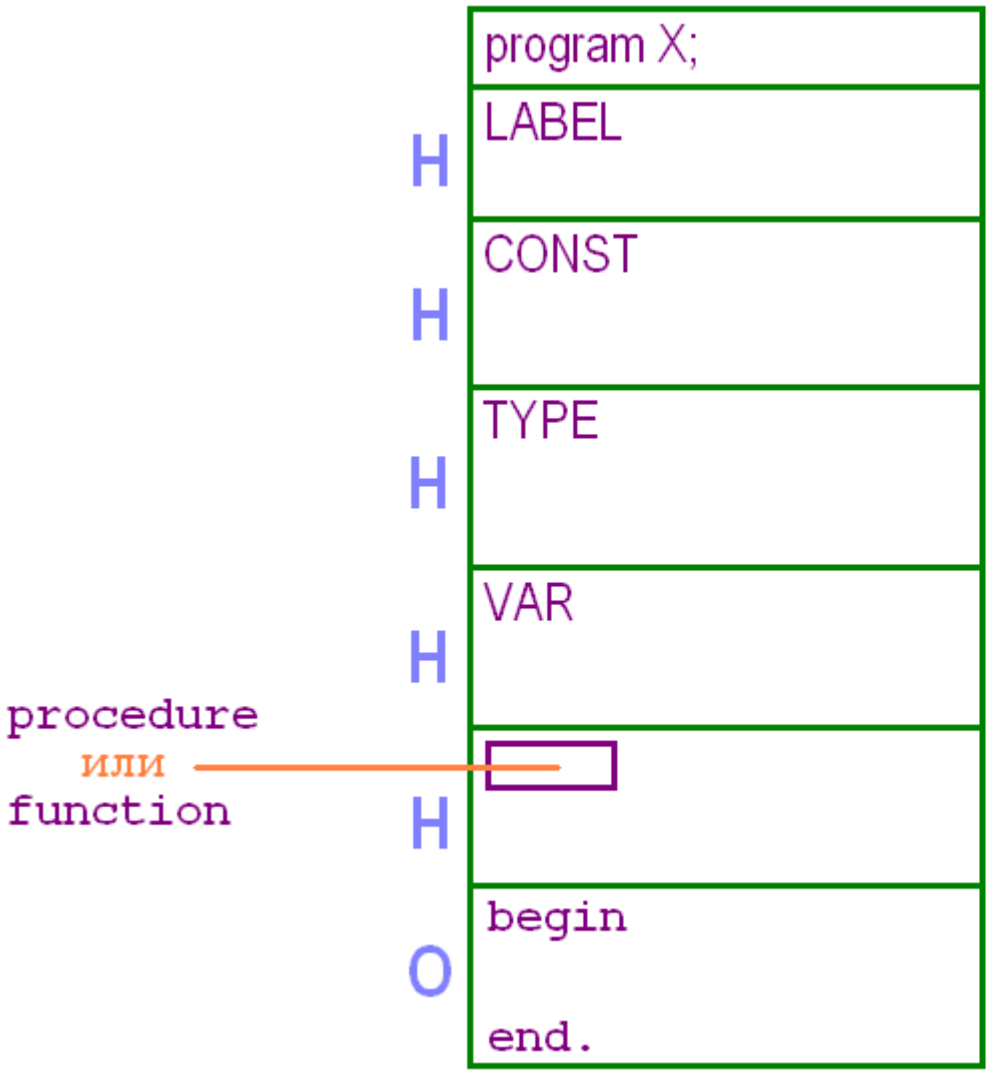
Программа

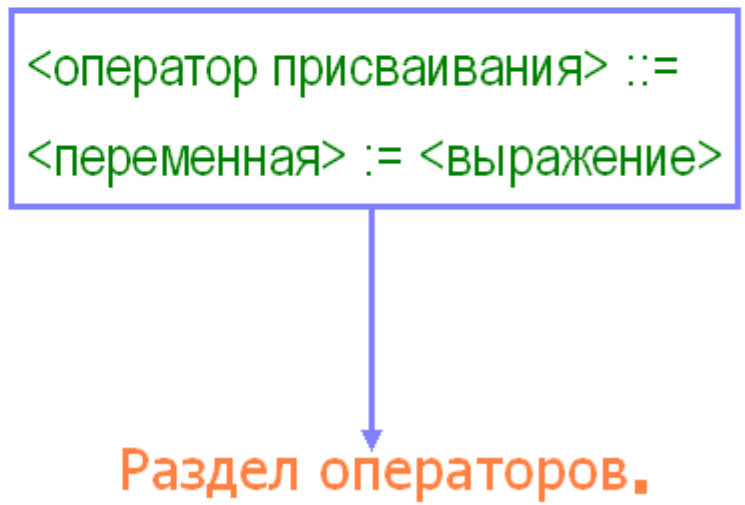
<программа> ::= <заголовок программы>;<блок>.

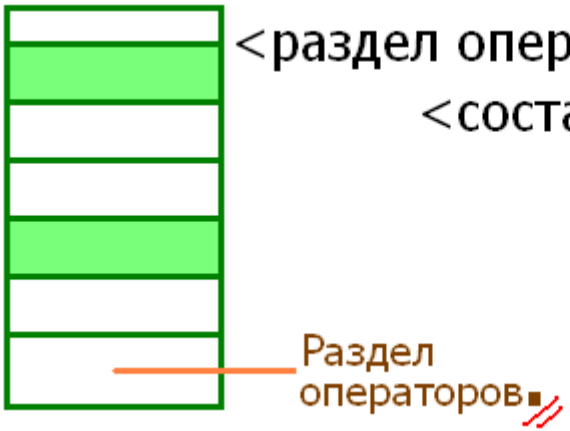
<блок> ::= <раздел меток>
<раздел констант>
<раздел типов>
<раздел переменных>
<раздел процедур и функций>
<раздел операторов>

<заголовок программы> ::= program <имя программы> |
program <имя программы>(<имя файла>{,<имя файла>})

Программа на языке ПАСКАЛЬ







<составной оператор>::=begin <оператор>{; <оператор>}end
Семантика: последовательное выполнение операторов,
если очередной оператор не изменяет посл-ть.
Прагматика: посл-ть операторов **=один оператор.**
begin I:=I+1; writeln(sin(I):8:4) end;

<пустой оператор>::=

<переменная>:= < выражение>

<выражение> –линейная запись математической формулы.

$$\frac{a + \sqrt{bc - 3a}}{6c}$$

```
(a + sqrt(b*c - 3*a))/(6*c)
```

$$d_{11}d_{22} - d_{12}d_{21}$$

```
d[1,1]*d[2,2] - d[1,2]*d[2,1]
```

простые переменные: a, b, c

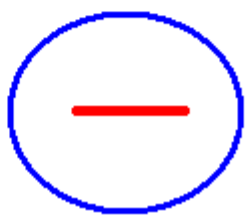
константы: 3, 6

переменные с индексами: d[1,1], d[2,2], d[1,2], d[2,1]

знаки арифм.операций: +,-,*,/

стандартные функции: sqrt

$$sign(x) = \begin{cases} +1, & \text{если } x > 0 \\ 0, & \text{если } x = 0 \\ -1, & \text{иначе} \end{cases}$$



<переменная>:= < выражение>

Переменная–элемент программы
–способный принимать **значения;**
–однозначно определяемый своим **наименованием.**
Наименование–синтаксическая конструкция
простом случае иначе
идентификатор**. . .**
Значение:
–имеется ровно одно в каждый момент времени;
–принадлежит заранее оговоренному множеству;
–хранится до замещения другим значением;
–может применяться неограниченное число раз;
–должно быть определено до первого применения.

Тип данных = множество значений и набор операций.

(А) Объявление переменных

(Б) Правила записи элементов

(В) Операции / Функции

Целый тип

(А) Переменные целого типа

```
var   I : integer;
var N,M : integer;
```

(Б) **<целое>** – синтаксис целых чисел: **00165**, **-998**.

Программная реализация – ∃ диапазон [**minInt .. maxInt**]

TP: **-32768 +32767**

(В.1) Операции над целыми

1. \+ **N + M** или +I
2. \- **N – M** или –I
3. * **N * M**
4. div **N div M** целая часть
5. mod **N mod M** остаток

Вещественный тип

(А) Переменные целого типа

```
var   R : real;
var X,Y : real;
```

(Б) **<вещественное>** – синтаксис вещ.: **1.5**, **-3.14**, **1E-20**.
Область значений определяется реализацией.

(В.1) Операции над вещественными: **+** | **-** | ***** | **/**

(В.2) Восемь стандартных функций: **REAL → REAL**

1. **abs(x)**	$\|x\|$	5. **exp(x)**	e^x
2. **sqr(x)**	x^2	6. **ln(x)**	$\ln(x)$
3. **sin(x)**	$\sin x$	7. **sqrt(x)**	$\sqrt{x}$
4. cos(x)	$\cos x$	8. **arctan(x)**	arctg(x)

Преобразования REAL ↔ INTEGER

Пусть
```
var X : real;
    N : integer;
```

REAL → INTEGER – функции **trunc** и **round**

N:=trunc(x)
отбрасывание дробной части

N:=round(x)
ближайшее целое

x	trunc(x)	round(x)
5.2	5	5
5.8	5	6
-5.2	-5	-5
-5.8	-5	-6

$$\text{round}(x) \overset{\text{def}}{=} \begin{cases} \text{trunc}(x+0.5), & \text{если } 0 \leqslant x \\ \text{trunc}(x-0.5), & \text{если } 0 > x \end{cases}$$

INTEGER → REAL – соглашения

- Целые на местах вещественных: **sin(N)**
- Правила определения типов операций: **X:=N**

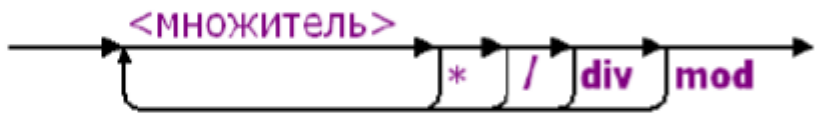
<переменная> := <выражение>

Арифметические выражения

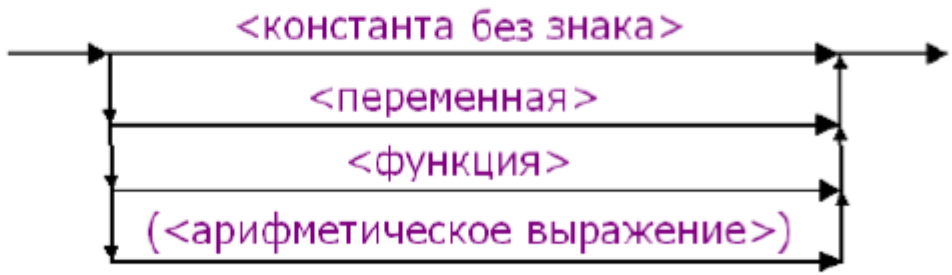
Тип выражения – тип результата. Арифм. выр. – integer или real.

<арифметическое выражение> ::=

<слагаемое> ::=



<множитель> ::=



константы без знака *функция*

```
+3*(maxInt/7+6.7) - abs(N+K)
```

(арифм. выр.) *переменные*

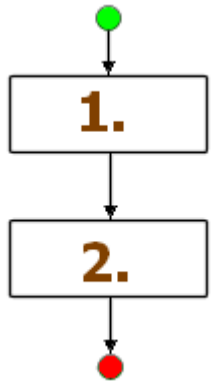
- Старшинство операций
- Правила определения типа выр.

<переменная>:= < выражение>

Семантика оператора присваивания

1.Вычислить **<выражение>.**

2.Полученное значение сделать значением переменной **<переменная>.**



Операторы: присваивания, ввода, вывода, пустой, составной, условный, перехода, цикла, . . .

Оператор ввода (с клавиатуры)

Read / Readln

Read(X_1, X_2, . . ., X_N);

Элемент списка ввода: char | integer | real

Пусть: var **X1** : integer; **X2** : char; **X3** : real;

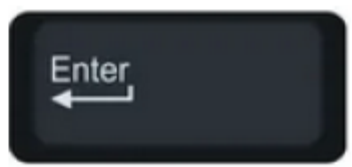
```
Read(X1,X2,X3);
```

Входной поток: `100␣A␣0.77␣`

Результат: `X1 = 100;  X2 = 'A'; X3 = 0.77`

```
Readln(X1,X2,X3);
```

Входной поток: `100␣A␣0.77<Enter>`



Оператор вывода (на экран)

write / writeln

write(X_1, X_2, . . . , X_N); ≡ write(X_1); write(X_2); . . . ; write(X_N);

real | integer | boolean | char + строки

Вместо write(X) можно использовать write(X:<целое>$_1$) и

(если X – вещественное) write(X:<целое>$_1$:< целое>$_2$)

ширина поля

к-во знаков в дробной

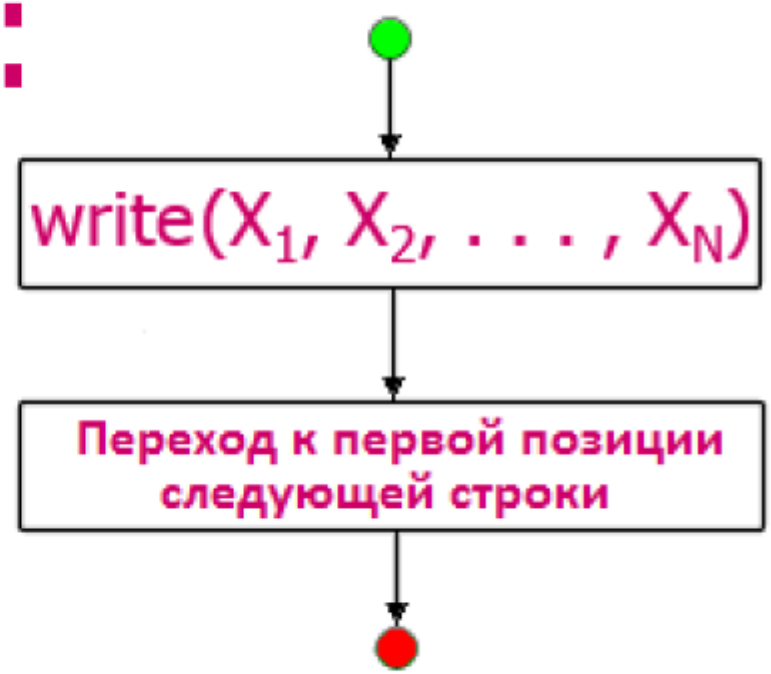
```
R:=0.00025;
writeln(R);
writeln(R:10:5);
writeln(R:10:4);
writeln(R:10:2);
```

```
␣2.5000000000E-04
␣␣␣0.00025
␣␣␣␣0.0003
␣␣␣␣␣␣0.00
```

Соглашение: write('И', 'в', 'а', 'н') = write('Иван')

Оператор вывода (на экран) - 2

writeln(X_1, X_2, . . . , X_N) :



Допускается оператор writeln без аргументов

Структура Си-программы: Си-программа состоит из одного или нескольких программных файлов (модулей, единиц компиляции).
Структура каждого программного файла:
объявления глобальных переменных
int main(список параметров); {
последовательность операторов
}
f1(список параметров); {
последовательность операторов
}
f2(список параметров); {
последовательность операторов
}
.....
fN(список параметров); {
последовательность операторов
}
Кроме того, программный файл может содержать инструкции препроцессора: #include, #define и др

Бзовые типы данных:
char (символьный),
int (целый),
float (с плавающей точкой),
double (двойной точности),
void (без значения).

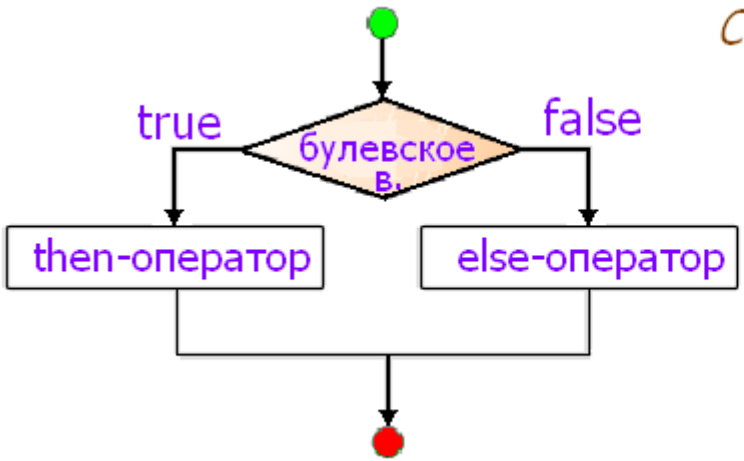
Переменная = тип + имя (идентификатор) + значение (оно, вообще говоря, изменяется при выполнении программы); доступ к текущему значению переменной осуществляет□ся по ее имени (либо по указателю). Каждая переменная является объектом программы. Ключевые слова (в стандарте C89 их 32, в стандарте C99 добавлено еще 5; кроме того «почти ключевым» является слово main могут использоваться только как ключевые слова, и не могут быть именами переменных. Тип переменной определяет интерпретацию операций, операндом которых является эта переменная, и тем самым определяет тип соответствующего выражения.

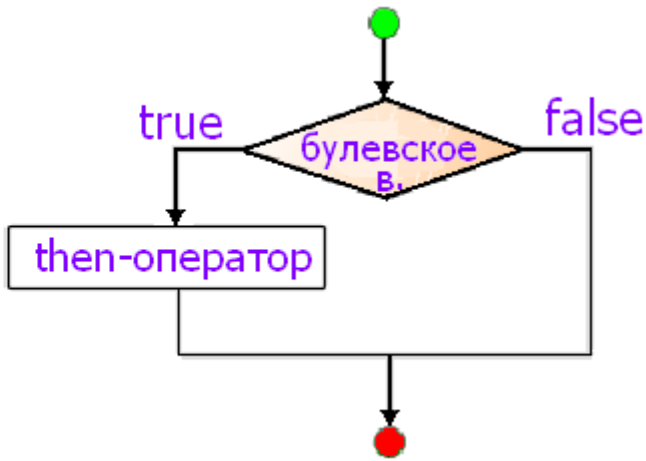
В стандарте ANSI C'89 определены следующие ключевые слова:
auto double int struct break else long switch case enum register typedef char extern return union const float short unsigned continue for signed void default goto sizeof volatile do if static while

Условный оператор в языке Паскаль

<условный оператор> ::=
<полный условный оп-р> **|<сокращенный условный оп-р>**
<полный условный оп-р>::=
if<булевское выр.> then<оператор> else<оператор>
<сокращенный условный оп-р> ::=
if<булевское выр.> then<оператор>

Семантика





Условный оператор в языке Си

Условный оператор (полный):

```
if (expr) { stmt } else { stmt } ;
```

или

```
if (expr) stmt; else stmt;
```

Условный оператор (краткий):

```
if (expr) { stmt ;
```

или

```
if (expr) stmt;
```

else всегда относится к ближайшему if:

```
if (x > 2)
    if (y > z)
        y = z;
    else
        z = y;
```

≡

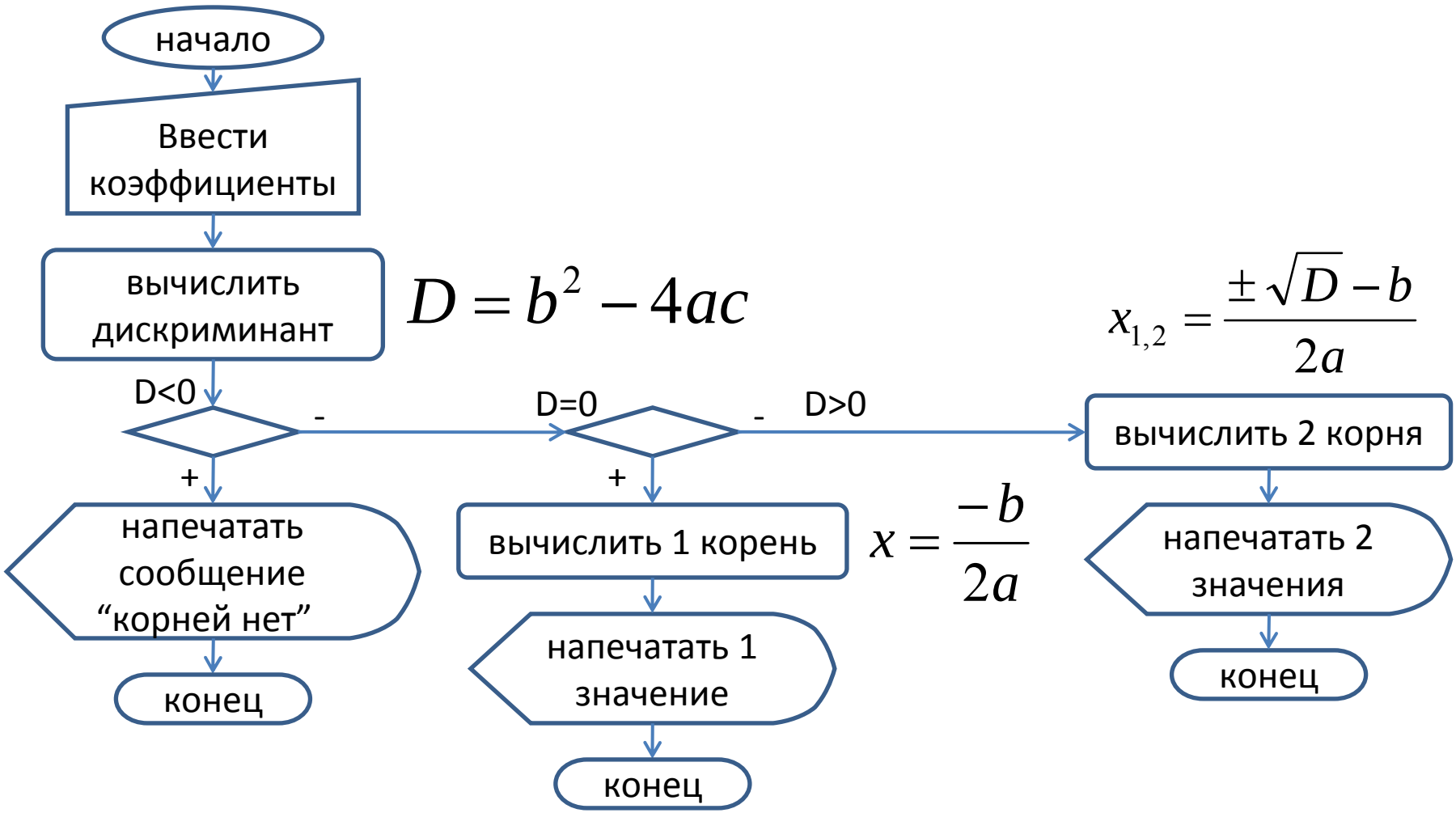
```
if (x > 2) {
    if (y > z) {
        y = z;
    } else {
        z = y
    }
};
```

≠

```
if (x > 2) {
    if (y > z)
        y = z;
} else
    z = y;
```

Задача.

Напечатать значения всех корней квадратного уравнения $ax^2+bx+c=0$, если они есть, или сообщение “корней нет”.



Задача.

Напечатать значения всех корней квадратного уравнения $ax^2+bx+c=0$, если они есть, или сообщение “корней нет”.

Решение на языке Паскаль

```
program sqvRoots(input, output);
var
   a, b, c, D: real;
begin
   read( a, b, c );
   D:=sqr(b)  - 4*a*c );
   if D<0 then
      writeln(”корней нет”)
   else
      if D=0 then
         writeln(”x=”,  -b/(2*a), )
      else
         writeln('x1=',  (-b+sqrt(D))/(2*a), '  x2=', (-b-sqrt(D))/(2*a)  );
end.
```

$$D = b^2 - 4ac$$

$$x = \frac{-b}{2a}$$

$$x_{1,2} = \frac{\pm\sqrt{D} - b}{2a}$$

Задача.

Напечатать значения всех корней квадратного уравнения $ax^2+bx+c=0$, если они есть, или сообщение “корней нет”.

Решение на языке Си

```
#include <stdio.h>
#include <math.h>
int main()
{
   float a,b,c,D ;
   scanf(“%f%f%f", &a,&b,&c);
 D = b*b - 4*a*c ;
 if (D<0)
    printf("корней нет \n");
 else
    if (D==0 )
       printf(" x=%f \n",  -b/(2*a) );
    else
       printf("x1=%f x2=%f \n“, (-b+d)/(2*a), (-b-d)/(2*a) );
}
```

$$D = b^2 - 4ac$$

$$x = \frac{-b}{2a}$$

$$x_{1,2} = \frac{\pm\sqrt{D} - b}{2a}$$

Другой пример Си-программы.

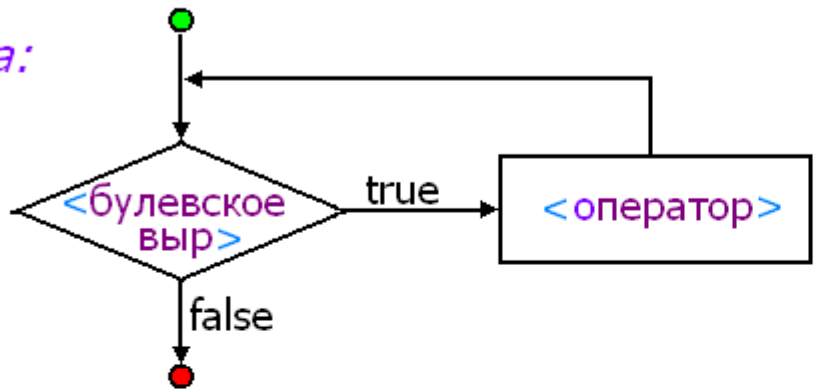
```
 /* "Магическое" число */
#include <stdio.h >
#include <stdlib.h>
int main(){
  int magic; /* "магическое" число */
  int guess;   /* угаданное число */

  magic = rand() /* генерация "магического" числа */

  printf("Угадай магическое число: ")
  scanf("%d", &guess);
  if (guess == magic) {
      printf("**Угадал**");
      printf("Магическое число равно %d\n", magic);
  }
  else {
      printf("**Не угадал**");
      if (guess > magic)  printf("Слишком большое.\n");
      else printf("Слишком маленькое.\n");
  }
  return 0;
}
```

Оператор цикла с предусловием в языке Паскаль

<оператор цикла с предусловием> ::=

while <булевское выр> do <оператор>

Семантика:



Задача. Вычислить min { k | k^2 > M}, M – задано.

Алгоритм.

```
K:=0;
while sqr(K) <= M do K:=K+1;
writeln('K=',K);
```

M=10:

K		sqr(K)<=M
0	0 <= 10	true
1	1 <= 10	true
2	4 <= 10	true
3	9 <= 10	true
4	16 <= 10	false

⇒ K=4

Оператор цикла с предусловием в языке Си

Цикл while:
 while (expression) stmt;

while (expression) { stmt }

Операторы break и continue: выход из внутреннего цикла и переход на следующую итерацию

Задача.

С точностью 10^{-5} вычислить x - наименьший положительный корень уравнения $\operatorname{tg} x = x$, используя метод деления отрезка пополам.

Задача.

С точностью 10^{-5} вычислить x - наименьший положительный корень уравнения $\operatorname{tg} x = x$, используя метод деления отрезка пополам.

Решение на языке Паскаль

```
program TgRoots(input, output);
var
   l, r, x: real;
begin
     {поиск корня на отрезке [l, r]=[π, 3π/2-ε]:  tg(l)<l, tg(r)>r}
          l:=3.14; r:=4.71;
          while r-l < 1e-5  do begin
               x:=(l+r)/2; {середина отрезка [l, r]}
               if sin(x)/cos(x)<x  then l:=x   { [l, r]:=[x, r] }
                                          else r:=x    { [l, r]:=[l, x] }
           end;
          x:=(l+r)/2 ;
          writeln('x=',  x  ),
end.
```

Задача.

С точностью 10^{-5} вычислить *x* - наименьший положительный корень уравнения tg *x* = *x,* используя метод деления отрезка пополам.

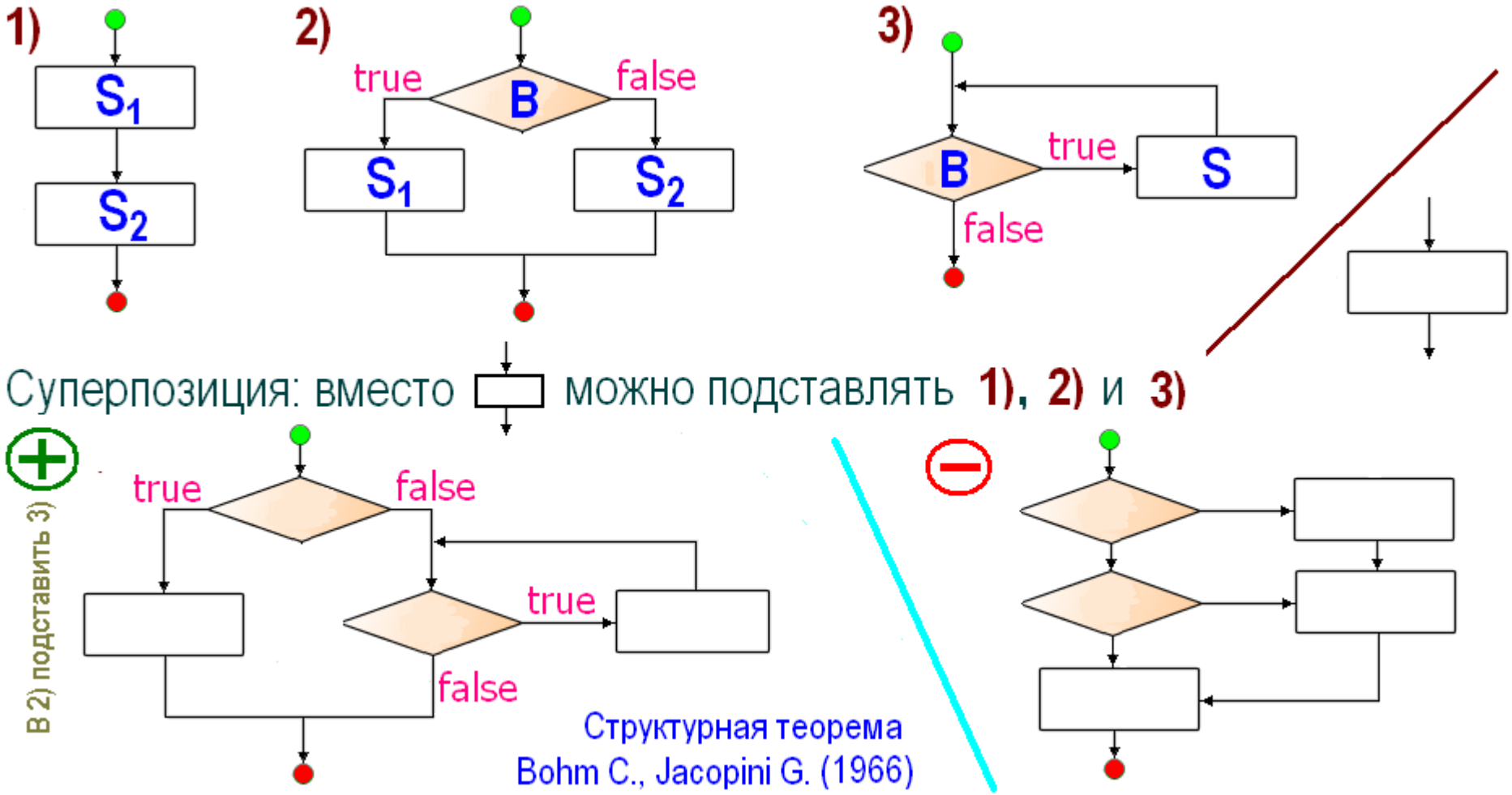
Решение на языке Си

```
#include <stdio.h>
#include <math.h> /* требуется параметр компиляции -lm */
int main()
{
    float  l, r, x;
    /* поиск корня на отрезке [l, r]=[π, 3π/2-ε]:  tg(l)<l, tg(r)>r */
     l=3.14; r=4.71;
    while ( r-l < 1e-5 )   {
         x=(l+r)/2;              /* середина отрезка [l, r] */
         if ( sin(x)/cos(x) < x )    l=x  ;   /*[l, r] →[x, r] */
         else                        r=x     /* [l, r]→[l, x]  */
     }
     x=(l+r)/2 ;
   printf("x=%f ", x  );
}
```

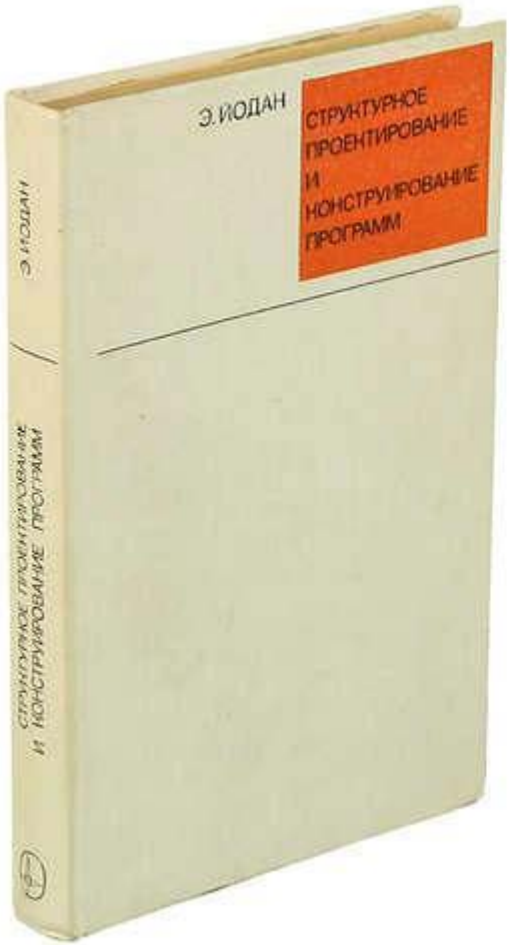
X=3.925000

Структурное программирование

дисциплина записи программ с использованием трех конструкций с одним входом и одним выходом



Структурное программирование



Йодан Э.
Структурное проектирование
и конструирование программ.
– М.: Мир, 1979.