Программирование на языке Си

Тема 1. Введение

Алгоритм

Алгоритм – это четко определенный план действий для исполнителя.

Свойства алгоритма

- дискретность: состоит из отдельных шагов (команд)
- понятность: должен включать только команды, известные исполнителю (входящие в СКИ)
- определенность: при одинаковых исходных данных всегда выдает один и тот же результат
- конечность: заканчивается за конечное число шагов
- массовость: может применяться многократно при различных исходных данных
- корректность: дает верное решение при любых допустимых исходных данных

Программа

Программа – это

- алгоритм, записанный на каком-либо языке программирования
- набор команд для компьютера

Команда – это описание действий, которые должен выполнить компьютер.

- откуда взять исходные данные?
- что нужно с ними сделать?
- куда поместить результат?

Языки программирования

- Машинно-ориентированные (низкого уровня) - каждая команда соответствует одной команде процессора (ассемблер)
- Языки высокого уровня – приближены к естественному (английскому) языку, легче воспринимаются человеком, не зависят от конкретного компьютера
  - *для обучения*: Бейсик, ЛОГО, Паскаль
  - *профессиональные*: Си, Фортран, Паскаль
  - *для задач искусственного интеллекта*: *Пролог, ЛИСП*
  - *для Интернета*: *JavaScript, Java, Perl, PHP, ASP*

Язык Си

1972-1974 – К. Томпсон, Д. Ритчи

- высокая скорость работы программ
- много возможностей
- стал основой многих современных языков *(C++, C#, Javascript, Java, ActionScript, PHP)*

- много шансов сделать ошибку, которая не обнаруживается автоматически

Простейшая программа

главная (основная) программа всегда имеет имя ***main***

```
main()
{

}
```

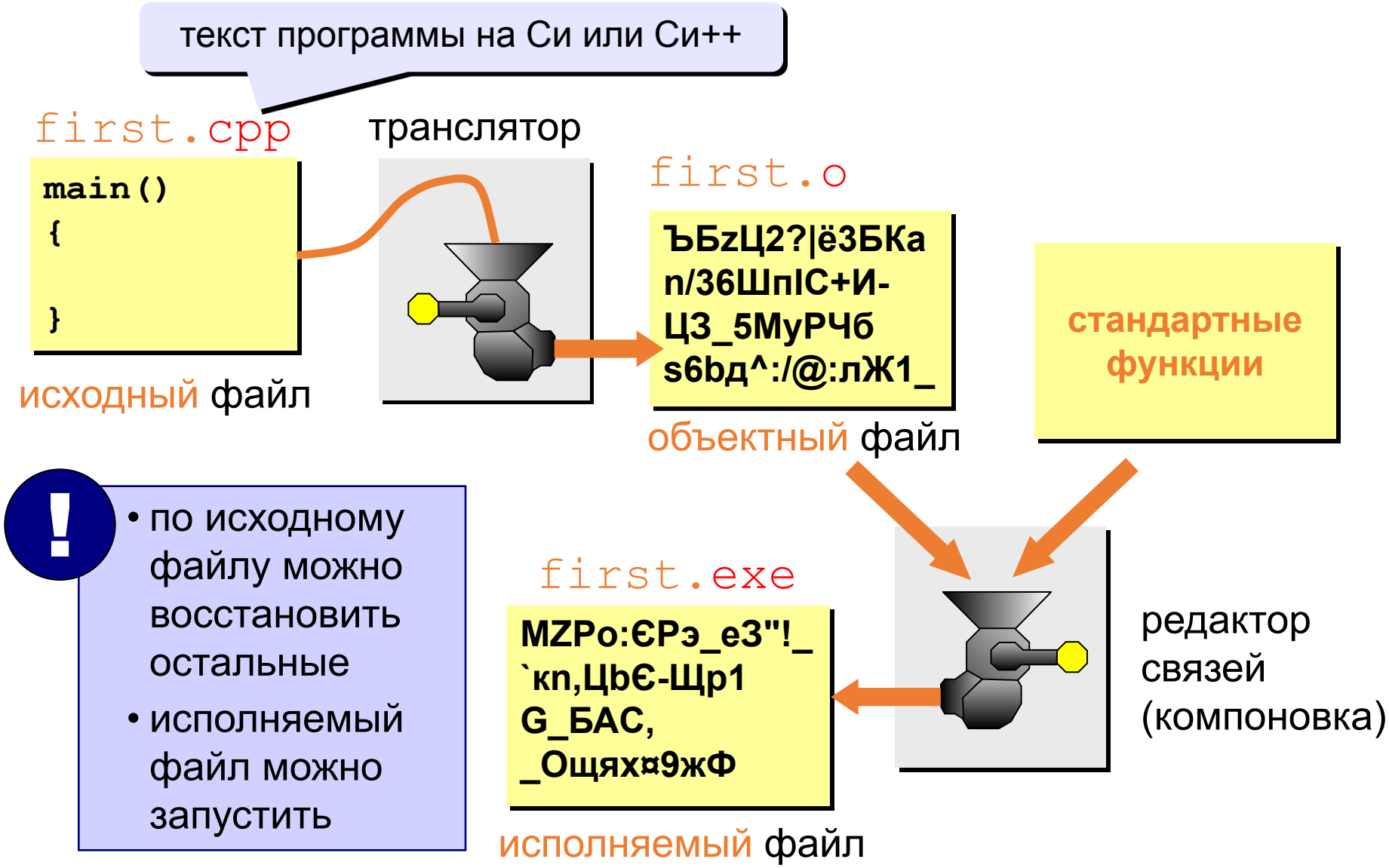
начало программы

«тело» программы (основная часть)

конец программы

? Что делает эта программа?

Что происходит дальше?



- по исходному файлу можно восстановить остальные
- исполняемый файл можно запустить

Вывод текста на экран

include = включить

файл ***stdio.h***: описание стандартных функций ввода и вывода

```
#include <stdio.h>
main()
{
printf("Привет!");
}
```

вызов стандартной функции ***printf = print format*** (форматный вывод)

этот текст будет на экране

Как начать работу?

Откр Закрыть

Файл Правк Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Создать Вставить Переключить Перейти

Проект Классы Отладка

privet.cpp

```
#include <stdio.h>
main()
{
printf("Privet!");
}
```

здесь мы набираем программу

Компилятор Ресурсы Журнал компиляции Отладка Результаты поиска Закрыть

Строка	Файл	Сообщение

сообщения об ошибках

Оболочка Dev C ++ 4.9

IDE = *Integrated Development Environment*

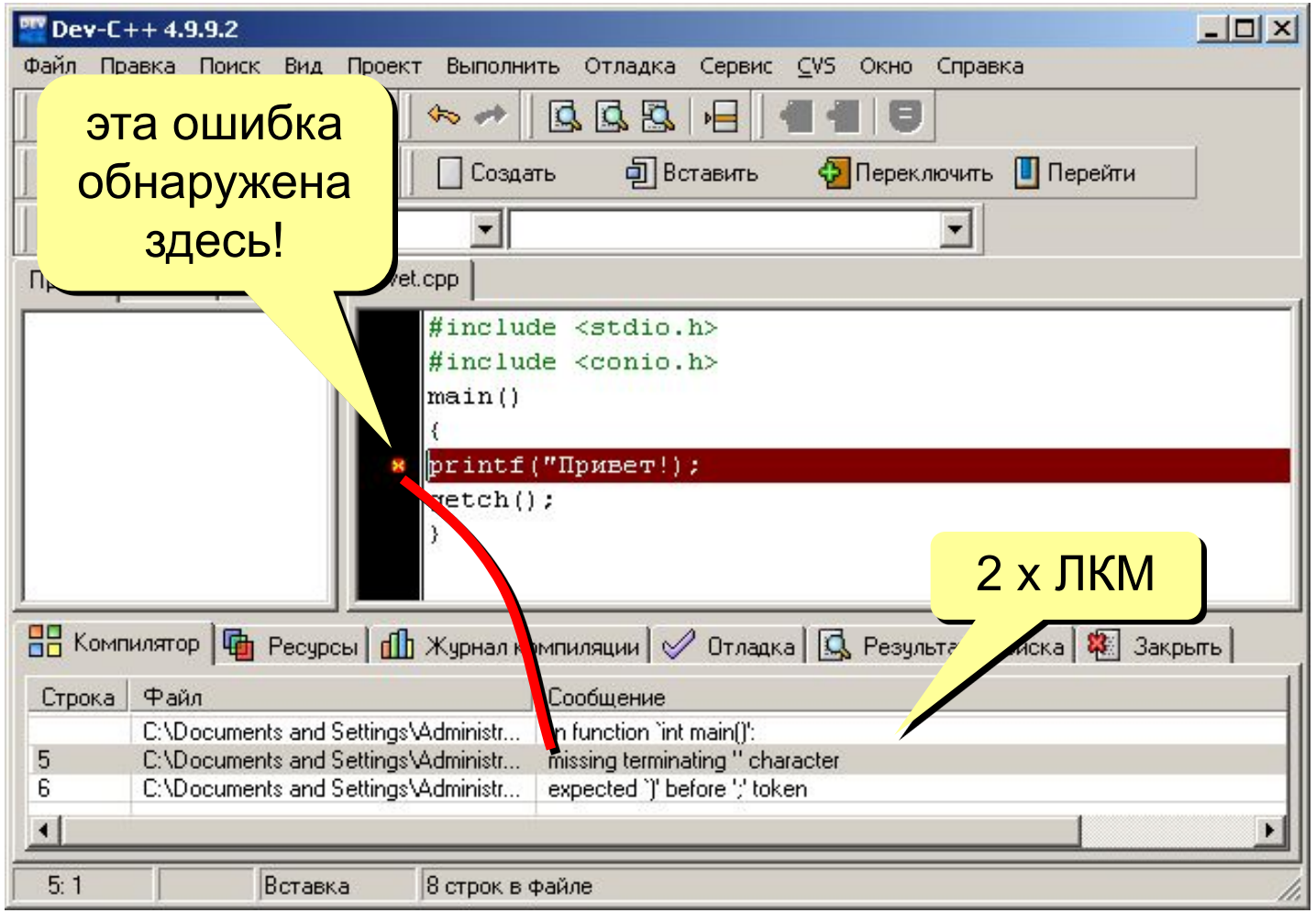
интегрированная среда разработки:

- текстовый редактор для создания и редактирования текстов программ
- транслятор для перевода текстов программ на Си и Си++ в команды процессора
- компоновщик для создания исполняемого файла (EXE-файла), подключаются стандартные функции
- отладчик для поиска ошибок в программах

Управление клавишами

Новый файл (Создать)	**Ctrl+N**	
Открыть файл	**Ctrl+O**	
Сохранить файл	**Ctrl+S**	
Закрыть окно с программой	**Ctrl-F4**	
Запуск программы	**F9**	
Отменить	**Ctrl-Z**	
Восстановить отмененное	**Shift-Ctrl-Z**	

Где ошибки?



! Ошибка может быть в конце предыдущей строки!

Наиболее «популярные» ошибки

xxx.h: No such file or directory	не найден заголовочный файл 'xxx.h' (неверно указано его имя, он удален или т.п.)
'xxx‘ undeclared (first use this function)	функция или переменная 'xxx' неизвестна
missing terminating " character	не закрыты кавычки "
expected ;	нет точки с запятой в конце оператора **в предыдущей строке**
expected }	не закрыта фигурная скобка

Ждем нажатия любой клавиши

```
#include <stdio.h>
#include <conio.h>
main()
{
printf("Привет!"); // вывод на экран
getch();   /* ждать нажатия клавиши */
}
```

файл ***conio.h***: описание функций для работы с клавиатурой и монитором

комментарий до конца строки

ждать нажатия на любую клавишу

комментарий между `/*` и `*/`

Переход на новую строку

```
#include <stdio.h>
#include <conio.h>
main()
{
printf("Привет,\n Вася!");
getch();
}
```

последовательность

\n (код 10)

переход на новую строку

на экране:

```
Привет,
Вася!
```

Задания

«4»: Вывести на экран текст "лесенкой"

```
Вася
    пошел
        гулять
```

«5»: Вывести на экран рисунок из букв

```
  Ж
 ЖЖЖ
ЖЖЖЖЖ
ЖЖЖЖЖЖЖ
HH HH
ZZZZZ
```

Программирование на языке Си

Тема 2. Переменные

© К.Ю. Поляков, 2007-2008

Что такое переменная?

Переменная – это ячейка в памяти компьютера, которая имеет имя и хранит некоторое значение.

- Значение переменной может меняться во время выполнения программы.
- При записи в ячейку нового значения старое стирается.

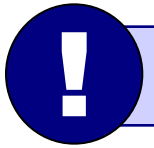
Типы переменных

- int – целое число (4 байта)
- float – вещественное число, *floating point* (4 байта)
- char – символ, *character* (1 байт)

Имена переменных

Могут включать

- латинские буквы (A-Z, a-z)
- знак подчеркивания _
- цифры 0-9



Имя не может начинаться с цифры!

НЕ могут включать

- русские буквы
- пробелы
- скобки, знаки +, =, !, ? и др.

Какие имена правильные?

AXby R&B 4Wheel Вася “PesBarbos”

TU154 [QuQu] _ABBA A+B

Объявление переменных

Объявить переменную = определить ее имя, тип, начальное значение, и выделить ей место в памяти.

```
main()
{
i
f
int Tu104,  186=23, Yak42;
float x=4.56, y, z;
char c, c2='A', m;
}
```

целая переменная **a**

вещественные перемен

целая и дробная части отделяются **точкой**

целые переменные **Tu104**, **Il86** и **Yak42**

вещественные переменные **x**, **y** и **z**
x = 4,56

символьные переменные **c**, **c2** и **m**
c2 = 'A'

! Если начальное значение не задано, в этой ячейке находится «мусор»!

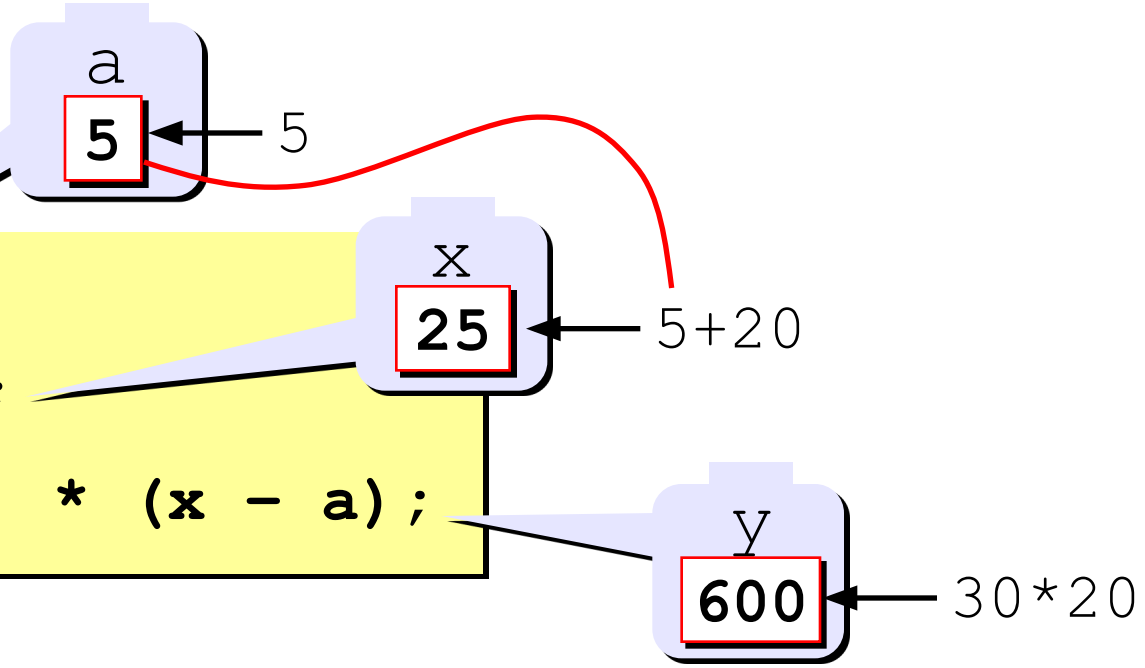
Оператор присваивания

Оператор – это команда языка программирования высокого уровня.

Оператор присваивания служит для изменения значения переменной.

Пример

```
a = 5;
x = a + 20;
y = (a + x) * (x - a);
```



Оператор присваивания

Общая структура:

куда записать

что

имя переменной = *выражение*;

Арифметическое выражение может включать

- константы (постоянные)
- имена переменных
- знаки арифметических операций:

+ – * / %

умножение

деление

остаток от деления

- вызовы функций
- круглые скобки ( )

? Для чего служат круглые скобки?

Какие операторы неправильные?

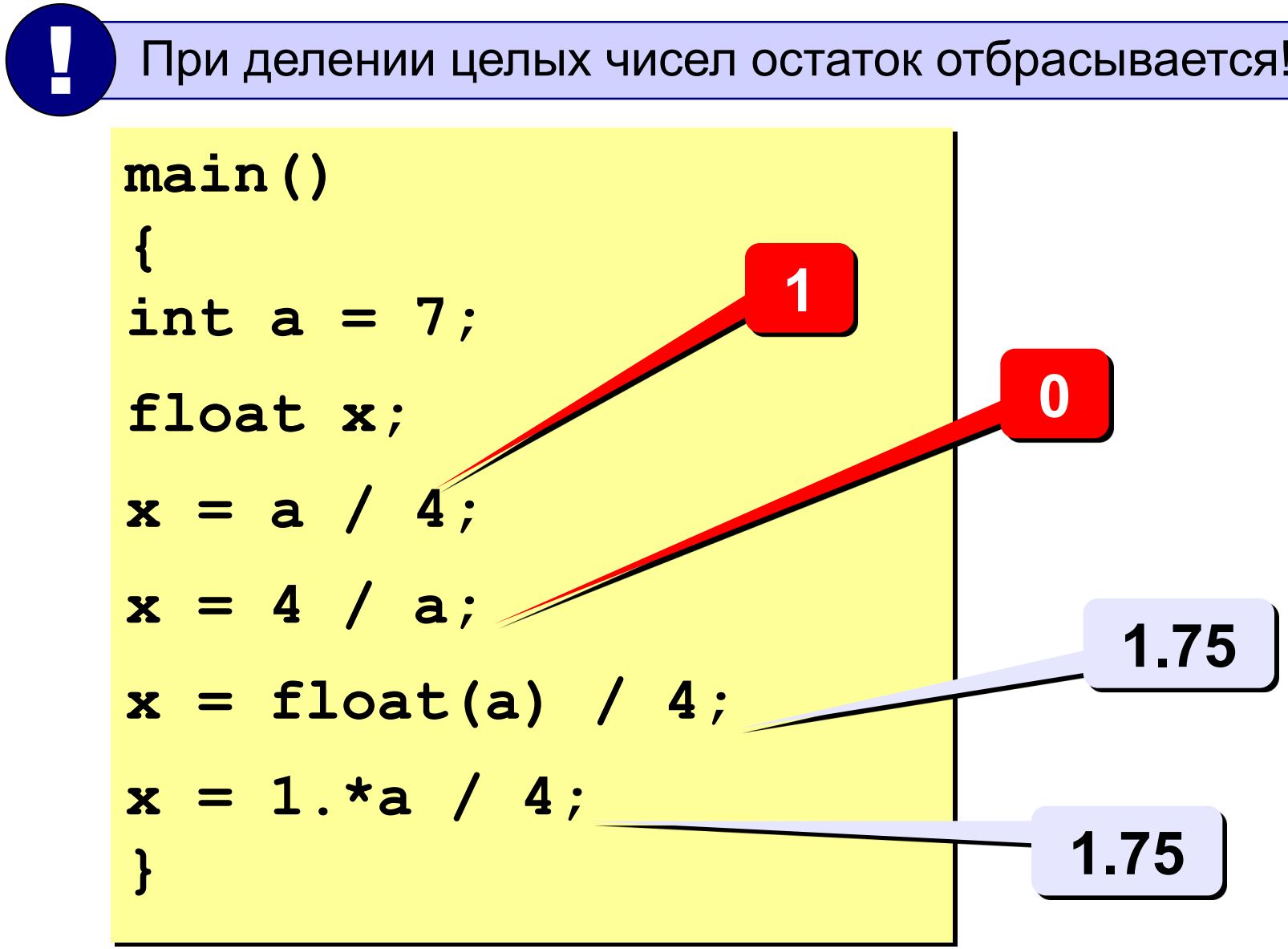
```
main()
 {
   int a, b;
   float  x, y;
   a = 5;
   10 = x;
   y = 7,8;
   b = 2.5;
   x = 2*(a + y);
   a = b + x;
 }
```

имя переменной должно быть **слева** от знака **=**

целая и дробная часть отделяются **точкой**

при записи вещественного значения в целую переменную **дробная часть будет отброшена**

Особенность деления в Си



Сокращенная запись операций в Си

полная запись	сокращенная запись
`a = a + 1;` инкремент	`a++;`
`a = a + b;`	`a += b;`
`a = a - 1;` декремент	`a--;`
`a = a - b;`	`a -= b;`
`a = a * b;`	`a *= b;`
`a = a / b;`	`a /= b;`
`a = a % b;`	`a %= b;`

Ручная прокрутка программы

```
main()
{
   int a, b;
   a = 5;
   b = a + 2;
   a = (a + 2)*(b - 3);
   b = a / 5;
   a = a % b;
   a++;
   b = (a + 14) % 7;
}
```

a	b
?	?
5	
	7
28	
	5
3	
4	
	4

Порядок выполнения операций

- вычисление выражений в скобках
- умножение, деление, % слева направо
- сложение и вычитание слева направо

```
        2  3  5  4   1     7  8   6   9
z  =  (5*a*c+3*(c-d))/a*(b-c)/  b;
```

$$z = \frac{5ac + 3(c-d)}{ab}(b-c)$$

$$x = \frac{a^2 + 5c^2 - d(a+b)}{(c+d)(d-2a)}$$

```
     2  6  3  4  7  5   1    12   8  11  10  9
x  =(a*a+5*c*c-d*(a+b))/((c+d)*(d-2*a));
```

Программирование на языке Си

Тема 3. Ввод и вывод

© К.Ю. Поляков, 2007-2008

Сложение двух чисел

Задача. Ввести два целых числа и вывести на экран их сумму.

Простейшее решение:

```
#include <stdio.h>
#include <conio.h>
main()
{
  int a, b, c;
  printf("Введите два целых чи
  scanf ("%d%d", &a, &b);
  c = a + b;
  printf("%d", c);
  getch();
}
```

подсказка для ввода

ввод двух чисел с клавиатуры

вывод результата

Ввод чисел с клавиатуры

scanf – форматный ввод

формат ввода

адреса ячеек, куда записать введенные числа

```
scanf ("%d%d", &a, &b);
```

Формат – символьная ст ка, которая показывает, какие числа вводятся (вывод ся).

- `%d` – целое число
- `%f` – вещественное по
- `%c` – 1 символ
- `%s` – символьная

&a – адрес переменной **a**

```
7652
12
```

a – значение переменной **a**

ждать ввода с клавиатуры двух целых чисел (через пробел или *Enter*), первое из них записать в переменную **a**, второе – в **b**

Что неправильно?

```
int a, b;
scanf ("%d", a);
scanf ("%d", &a, &b);
scanf ("%d%d", &a);

scanf ("%d %d", &a, &b);
scanf ("%f%f", &a, &b);
```

&a

%d%d

&a, &b

убрать пробел

%d%d

Вывод чисел на экран

здесь вывести целое число

это число взять из ячейки **c**

```
printf ("%d", c);
```

```
printf ("Результат: %d", c);
```

```
printf ("%d+%d=%d", a, b, c
);
```

формат вывода

список значений

```
printf ("%d+%d=%d", a, b, a+b
);
```

арифметическое выражение

Вывод целых чисел

```
int x = 1234;
printf ("%d", x);
```

или "%i"

```
1234
```

минимальное число позиций

или "%9i"

```
printf ("%9d", x);
```

```
     1234
```

всего 9 позиций

5 4

Вывод вещественных чисел

```
float x = 123.4567;
printf ("%f", x);
```

```
123.456700
```

минимальное число позиций, **6 цифр** в дробной части

```
printf ("%9.3f",
x);
```

```
  123.456
```

всего 9 позиций, **3 цифры** в дробной части

```
printf ("%e", x);
```

```
1.234560e+02
```

стандартный вид: $\mathbf{1{,}23456 \cdot 10^2}$

```
printf ("%10.2e", x);
```

```
  1.23e+02
```

всего 10 позиций, **2 цифры** в дробной части мантиссы

Полное решение

```
#include <stdio.h>
#include <conio.h>
main()
{
  int a, b, c;
  printf("Введите два целых числа\n");
  scanf("%d%d", &a, &b);
  c = a + b;
  printf("%d+%d=%d", a, b, c);
  getch();
}
```

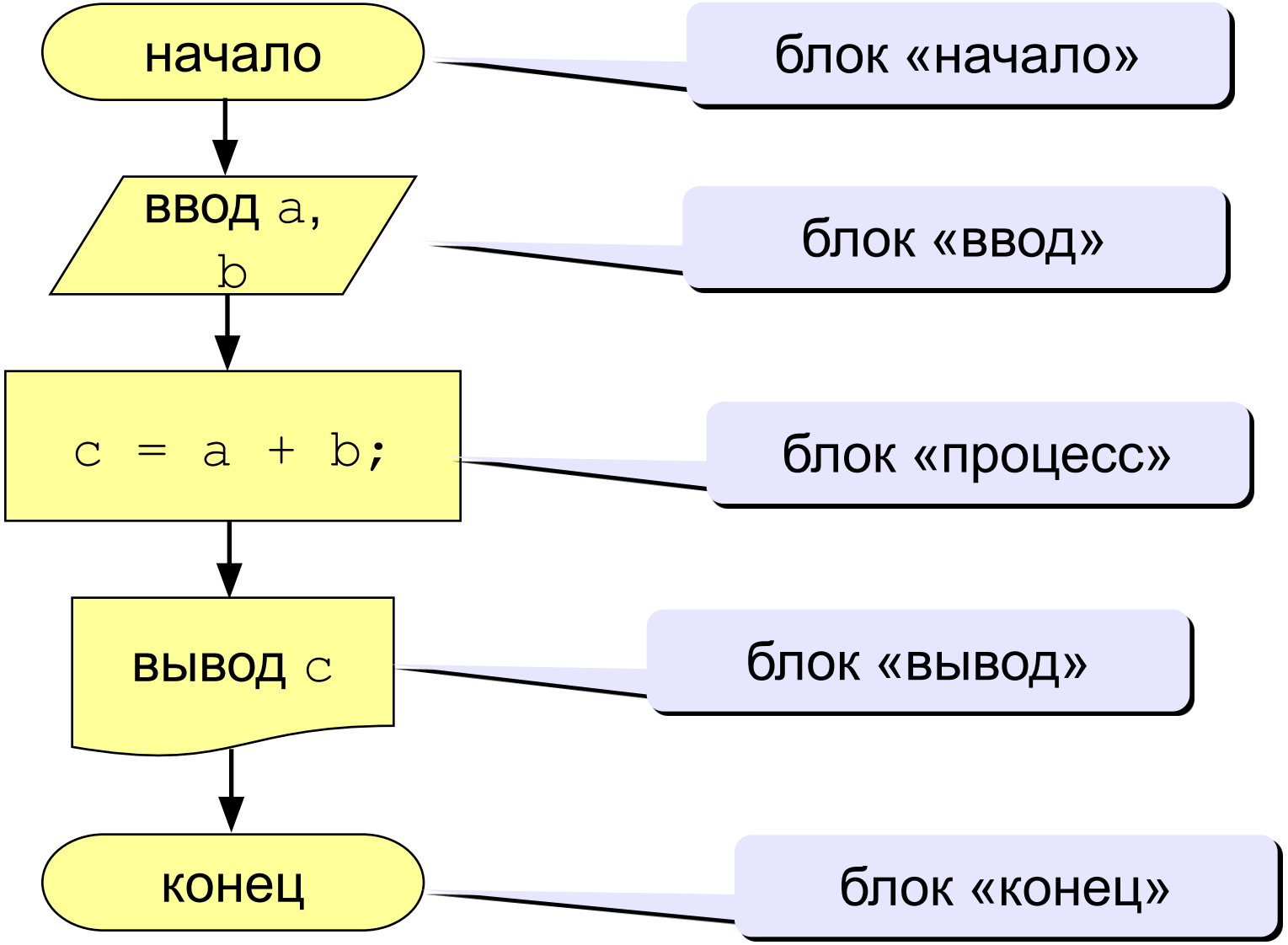
Протокол:

```
Введите два целых числа
25 30
25+30=55
```

это выводит компьютер

это вводит пользователь

Блок-схема линейного алгоритма



Задания

«4»: Ввести три числа, найти их сумму и произведение.

Пример:

```
Введите три числа:
4   5   7
4+5+7=16
4*5*7=140
```

«5»: Ввести три числа, найти их сумму, произведение и среднее арифметическое.

Пример:

```
    Введите три числа:
4   5   7
4+5+7=16
4*5*7=140
(4+5+7)/3=5.33
```

Программирование на языке Си

Тема 4. Ветвления

© К.Ю. Поляков, 2007-2008

Разветвляющиеся алгоритмы

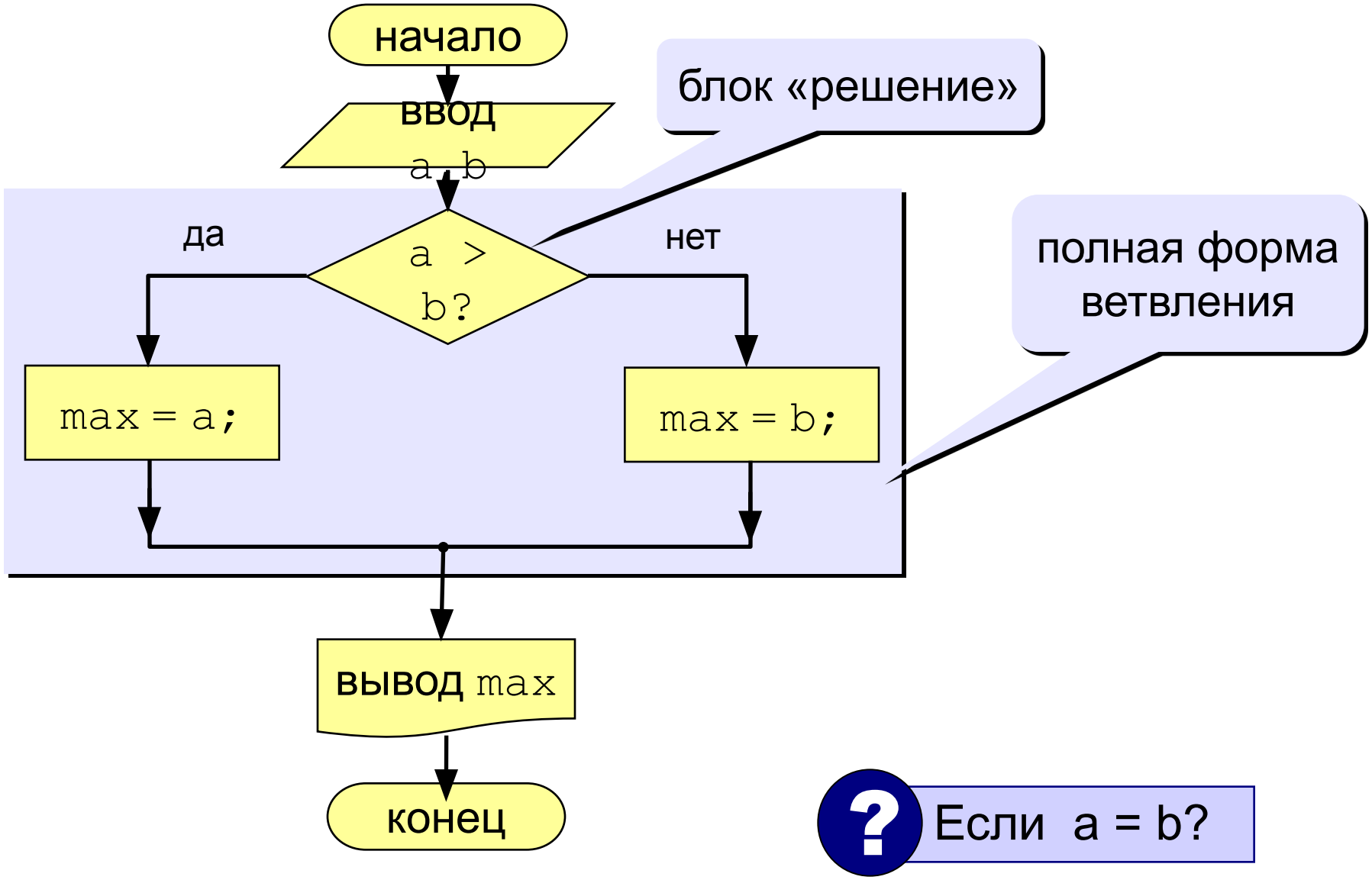
Задача. Ввести два целых числа и вывести на экран наибольшее из них.

Идея решения: надо вывести на экран первое число, если оно больше второго, или второе, если оно больше первого.

Особенность: действия исполнителя зависят от некоторых условий (*если … иначе …*).

Алгоритмы, в которых последовательность шагов зависит от выполнения некоторых условий, называются **разветвляющимися.**

Вариант 1. Блок-схема



Вариант 1. Программа

```
main()
{
    int a, b, max;
    printf("Введите два целых числа\n");
    scanf("%d%d", &a, &b );
    if (a > b) {
        max = a;
    }
    else {
        max = b;
    }
    printf("Наибольшее число %d", max);
}
```

полная форма условного оператора

Условный оператор

```
if ( условие )
   {
        // что делать, если условие верно
   }
else
   {
        // что делать, если условие неверно
   }
```

Особенности:

- вторая часть (`else` …) может отсутствовать (неполная форма)
- если в блоке один оператор, можно убрать { }

Что неправильно?

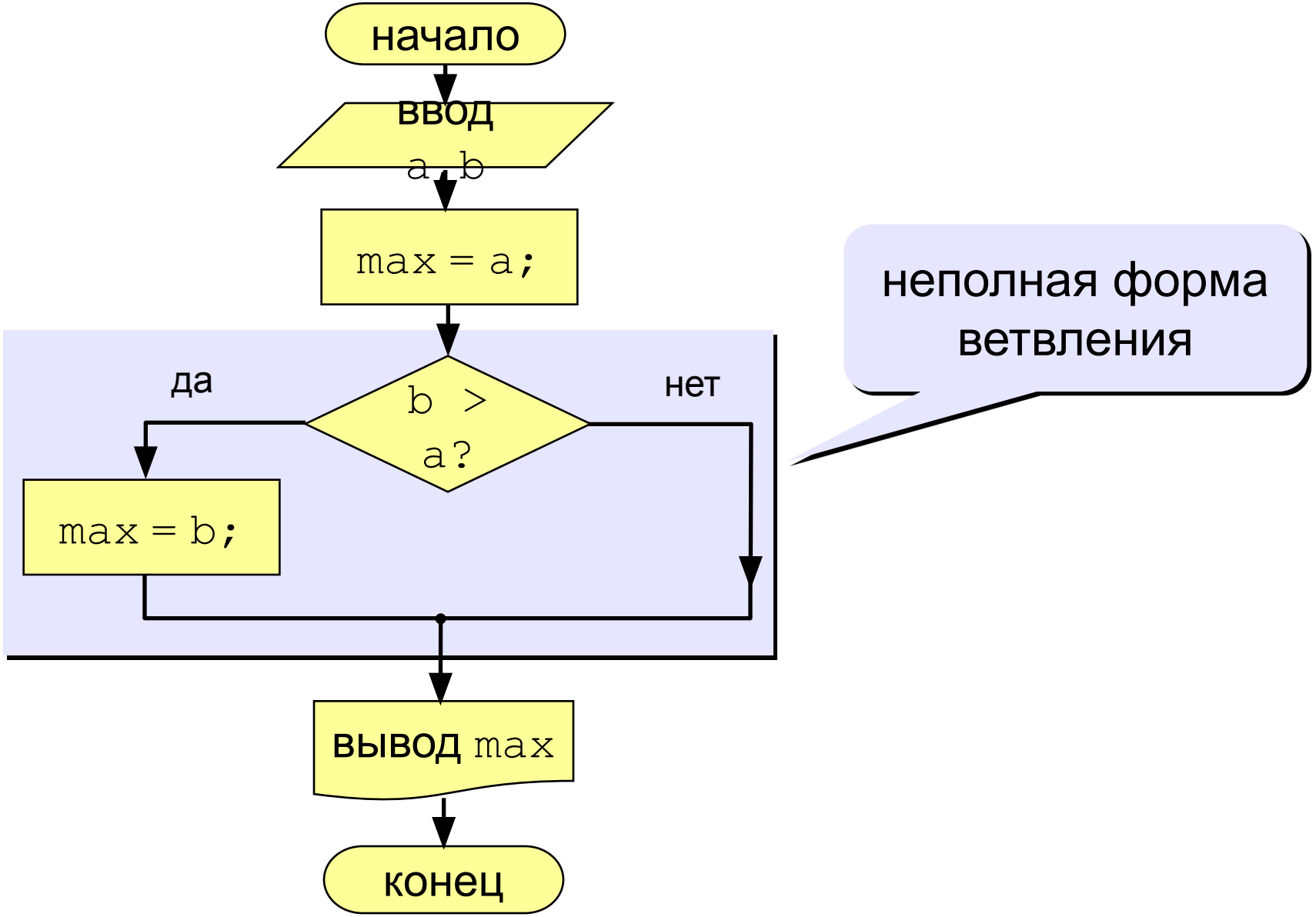
```
if ( a > b ) {
  a = b;
  }
else
  b = a;
```

```
if ( a > b ) {
  a = b; }
else
  b = a;
```

```
if ( a > b ) a = b;
else
  b = a;
```

```
if ( a > b ) {
 a = b;
 c = 2*a; }
else
 b = a;
```

Вариант 2. Блок-схема



Вариант 2. Программа

```
main()
{
   int a, b, max;
   printf("Введите два целых числа\n");
   scanf("%d%d", &a, &b );
   max = a;
   if (b > a)
     max = b;
   printf("Наибольшее число %d", max);
}
```

неполная форма условного оператора

Вариант 2Б. Программа

```
main()
{
   int a, b, max;
   printf("Введите два целых числа\n");
   scanf("%d%d", &a, &b );
   max = b;
   if ( a > b )
      max = a;
   printf("Наибольшее число %d", max);
}
```

Задания

«4»: Ввести три числа и найти наибольшее из них.

Пример:

```
Введите три числа:
4   15   9
Наибольшее число 15
```

«5»: Ввести пять чисел и найти наибольшее из них.

Пример:

```
Введите пять чисел:
4    15    9    56    4
Наибольшее число 56
```

Программирование на языке Си

Тема 5. Сложные условия

© К.Ю. Поляков, 2007-2008

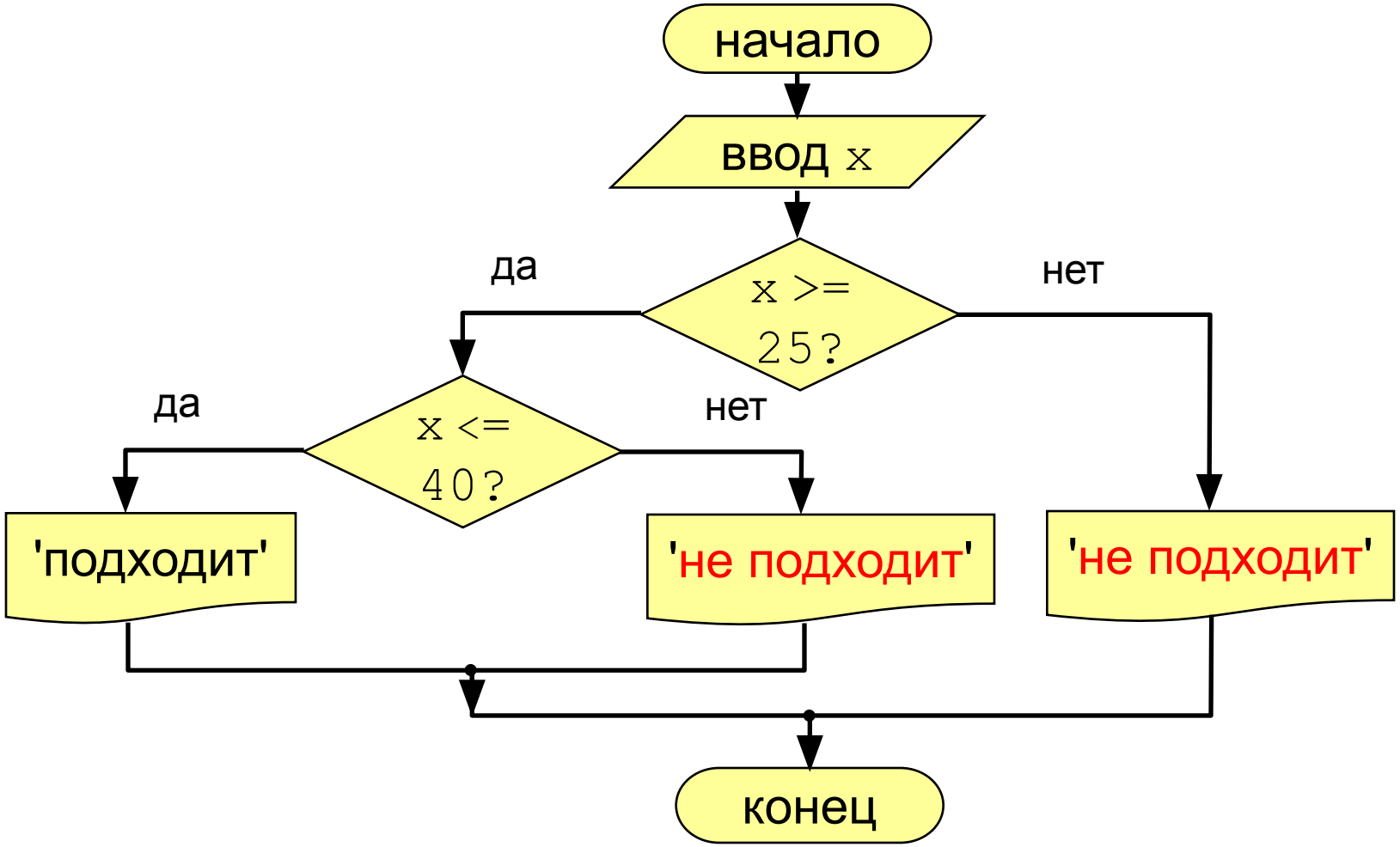
Сложные условия

Задача. Фирма набирает сотрудников от 25 до 40 лет включительно. Ввести возраст человека и определить, подходит ли он фирме (вывести ответ «подходит» или «не подходит»).

Особенность: надо проверить, выполняются ли два условия одновременно.

? Можно ли решить известными методами?

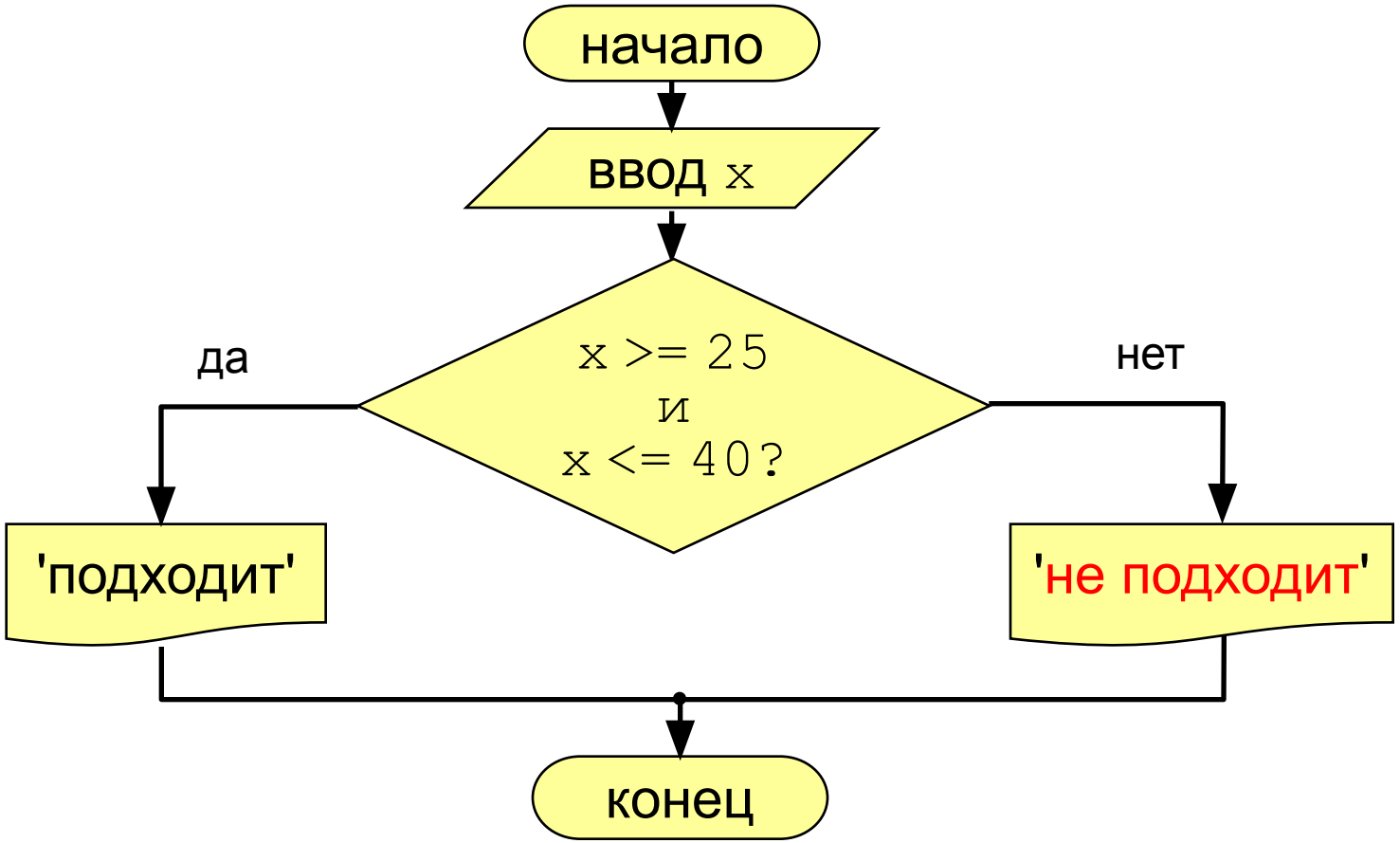
Вариант 1. Алгоритм



Вариант 1. Программа

```
main()
{
 int x;
 printf("Введите возраст\n");
 scanf("%d", &x);
 if (x >= 25)
   if (x <= 40)
        printf("Подходит");
   else printf("Не подходит");
 else
   printf("Не подходит");
}
```

Вариант 2. Алгоритм



Вариант 2. Программа

```
main()
{
int x;
printf("Введите возраст\n");
scanf("%d", &x);
if ( x >= 25 && x <= 40 )
     printf("Подходит");
else printf("Не подходит");
}
```

сложное условие

Сложные условия

Сложное условие – это условие, состоящее из нескольких простых условий (отношений), связанных с помощью логических операций:

- `!` – НЕ (*not*, отрицание, инверсия)
- `&&` – И (*and*, логическое умножение, конъюнкция, одновременное выполнение условий)
- `||` – ИЛИ (*or*, логическое сложение, дизъюнкция, выполнение хотя бы одного из условий)

Простые условия (отношения)

`<` `<=` `>` `>=` `==` `!=`

равно

не равно

Сложные условия

Порядок выполнения сложных условий:

- выражения в скобках
- `!` (НЕ, отрицание)
- `<, <=, >, >=`
- `==, !=`
- `&&` (И)
- `||` (ИЛИ)

Пример:

```
      2   1      6   3     5      4
if ( !(a > b) || c != d && b == a)
   {
   ...
   }
```

Сложные условия

Истинно или ложно при `a = 2; b = 3; c = 4;`

```
!(a > b)                    1
a < b && b < c              1
!(a >= b) || c == d         1
a < c || b < c && b < a     1
a > b || !(b < c)           0
```

Для каких значений `x` истинны условия:

```
x < 6 && x < 10
x < 6 && x > 10
x > 6 && x < 10
x > 6 && x > 10
x < 6 || x < 10
x < 6 || x > 10
x > 6 || x < 10
x > 6 || x > 10
```

$(-\infty, 6)$	$x < 6$
$\varnothing$	
$(6, 10)$	
$(10, \infty)$	$x > 10$
$(-\infty, 10)$	$x < 10$
$(-\infty, 6) \cup (10, \infty)$	
$(-\infty, \infty)$	
$(6, \infty)$	$x > 6$

Задания

«4»: Ввести номер месяца и вывести название времени года.

Пример:

```
Введите номер месяца:
4
весна
```

«5»: Ввести возраст человека (от 1 до 150 лет) и вывести его вместе с последующим словом «год», «года» или «лет».

Пример:

```
Введите возраст:
24
Вам 24 года
```

```
Введите возраст:
57
Вам 57 лет
```

Программирование на языке Си

Тема 6. Циклы

© К.Ю. Поляков, 2007-2008

Циклы

Цикл – это многократное выполнение одинаковой последовательности действий.

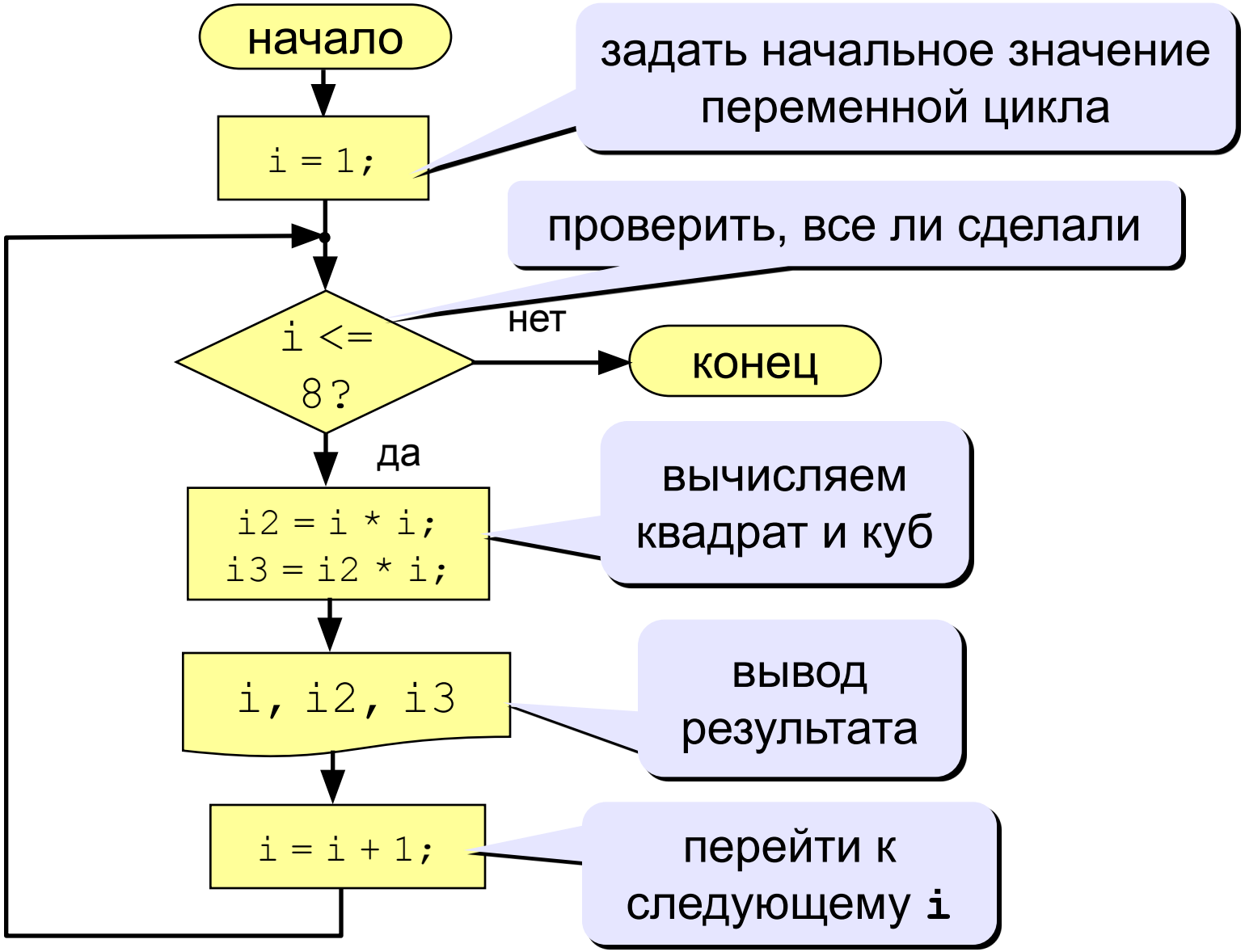
- цикл с известным числом шагов
- цикл с неизвестным числом шагов (цикл с условием)

Задача. Вывести на экран квадраты и кубы целых чисел от 1 до 8 (от `a` до `b`).

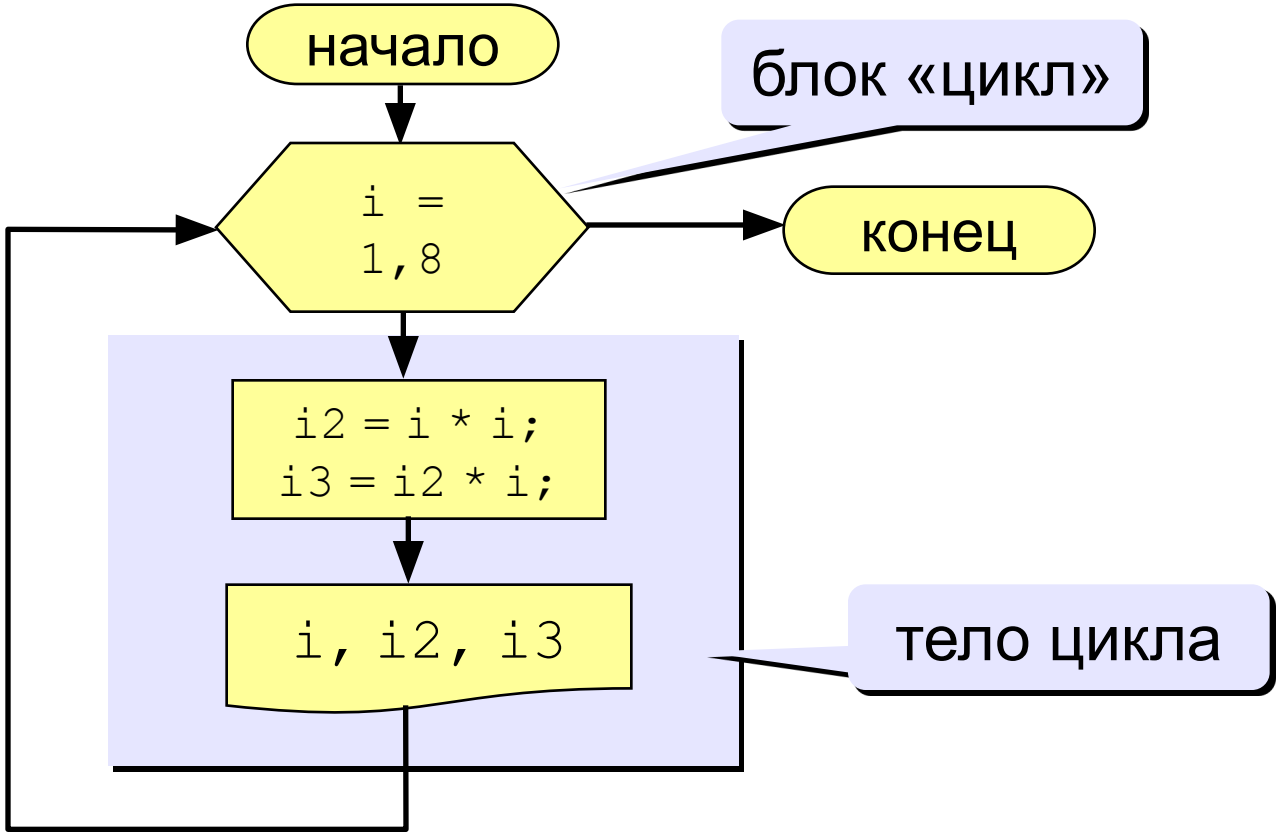
Особенность: одинаковые действия выполняются 8 раз.

? Можно ли решить известными методами?

Алгоритм


начало
i = 1;
задать начальное значение переменной цикла
проверить, все ли сделали
i <= 8?
нет
конец
да
i2 = i * i;
i3 = i2 * i;
вычисляем квадрат и куб
i, i2, i3
вывод результата
i = i + 1;
перейти к следующему i

Алгоритм (с блоком «цикл»)



Программа

```
main()
{
int i, i2, i3;
for (i=1; i<=8; i++)
  {
  i2 = i*i;
  i3 = i2*i;
  printf("%4d %4d %4d\n", i, i2,
  i3);
  }
}
```

начальное значение

заголовок цикла

конечное значение

цикл

переменная цикла

изменение на каждом шаге: **i=i+1**

начало цикла

тело цикла

конец цикла

ровные столбики

Цикл с уменьшением переменной

Задача. Вывести на экран квадраты и кубы целых чисел от 8 до 1 (в обратном порядке).

Особенность: переменная цикла должна уменьшаться.

Решение:

```
for ( i=8; i>=1; i-- )
  {
  i2 = i*i;
  i3 = i2*i;
  printf("%4d %4d %4d\n", i, i2, i3);
  }
```

Цикл с переменной

```
for (начальные значения;
     условие продолжения цикла;
     изменение на каждом шаге)
 {
 // тело цикла
 }
```

Примеры:

```
for (a=2; a<b; a+=2) { ... }
```

```
for (a=2, b=4; a<b; a+=2) { ... }
```

```
for (a=1; c<d; x++) { ... }
```

```
for (; c<d; x++) { ... }
```

```
for (; c<d; ) { ... }
```

Цикл с переменной

Особенности:

- *условие* проверяется в начале очередного шага цикла, если оно ложно цикл не выполняется;
- *изменения* (третья часть в заголовке) выполняются в конце очередного шага цикла;
- если *условие* никогда не станет ложным, цикл может продолжаться бесконечно (зацикливание)

```
for(i=1; i<8; i++) { i--; }
```

! Не рекомендуется менять переменную цикла в теле цикла!

- если в теле цикла один оператор, скобки {} можно не ставить:

```
for(i=1; i<8; i++) a+=b;
```

Цикл с переменной

Особенности:

- после выполнения цикла во многих системах устанавливается первое значение переменной цикла, при котором нарушено условие:

```
for (i=1; i<=8; i++)
  printf("Привет");
printf("i=%d", i);
```

i=9

```
for (i=8; i>=1; i--)
  printf("Привет");
printf("i=%d", i);
```

i=0

Сколько раз выполняется цикл?

```
a=1;
for(i=1; i<4; i++) a++;
```

a = 4

```
a=1;
for(i=1; i<4; i++) a=a+i;
```

a = 7

```
a=1; b=2;
for(i=3; i>=1; i--)a+=b;
```

a = 7

```
a=1;
for(i=1; i>=3; i--)a=a+1;
```

a = 1

```
a=1;
for(i=1; i<=4; i--)a++;
```

зацикливание

Задания

«4»: Ввести `a` и `b` и вывести квадраты и кубы чисел от `a` до `b`.

Пример:

```
Введите границы интервала:
4 6
  4   16    64
  5   25   125
  6   36   216
```

«5»: Вывести квадраты и кубы 10 чисел следующей последовательности: 1, 2, 4, 7, 11, 16, …

Пример:

```
  1      1      1
  2      4      8
  4     16     64
       ...
 46   2116  97336
```

Программирование на языке Си

Тема 7. Циклы с условием

© К.Ю. Поляков, 2007-2008

Цикл с неизвестным числом шагов

Пример: Отпилить полено от бревна. Сколько раз надо сделать движения пилой?

Задача: Ввести целое число (<2000000) и определить число цифр в нем.

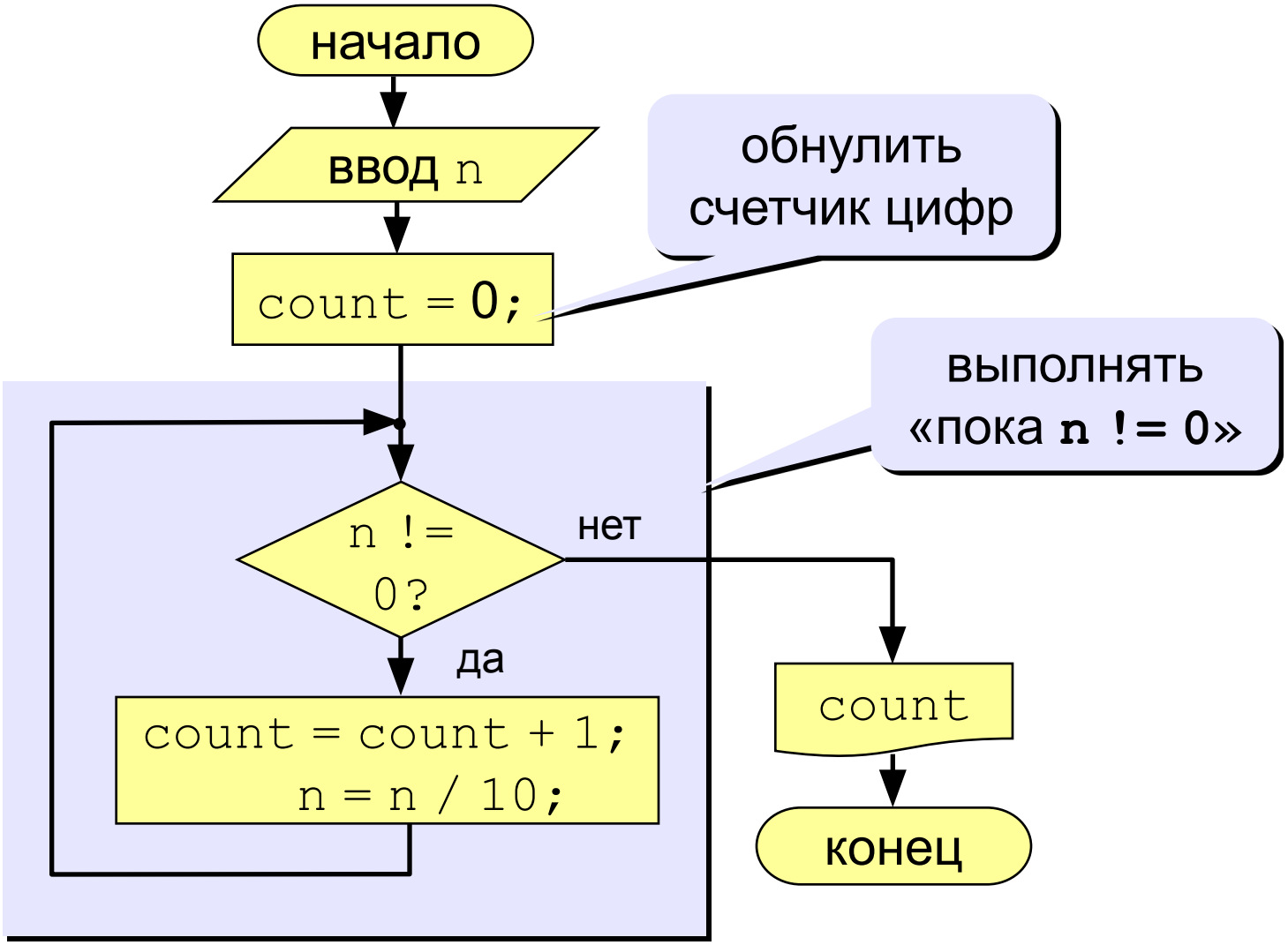
Идея решения: Отсекаем последовательно последнюю цифру, увеличиваем счетчик.

n	count
123	0
12	1
1	2
0	3

Проблема: Неизвестно, сколько шагов надо сделать.

Решение: Надо остановиться, когда `n = 0`, т.е. надо делать «пока `n != 0`».

Алгоритм


начало
ввод n
count = 0;
обнулить
счетчик цифр
выполнять
«пока n != 0»
n !=
0?
нет
да
count = count + 1;
n = n / 10;
count
конец

Программа

```
main()
{
int n, count, n1;
printf("Введите целое число\n");
scanf("%d", &n);
count = 0; n1 = n;
while (n != 0)
   {
   count ++;
   n = n / 10;
   }
printf("В числе %d нашли %d цифр", n1, count);
}
```

выполнять «пока **n != 0**»

? Что плохо?

Цикл с условием

```
while ( условие )
   {
   // тело цикла
   }
```

Особенности:

- можно использовать сложные условия:

```
while (a<b && b<c ) { ... }
```

- если в теле цикла только один оператор, скобки { } можно не писать:

```
while ( a < b ) a++;
```

Цикл с условием

Особенности:

- условие пересчитывается каждый раз при входе в цикл
- если условие на входе в цикл ложно, цикл не выполняется ни разу

```
a = 4; b = 6;
while ( a > b ) a=a-b;
```

- если условие никогда не станет ложным, программа зацикливается

```
a = 4; b = 6;
while ( a<b ) d=a+b;
```

Сколько раз выполняется цикл?

```
a=4; b=6;
while(a<b) a++;
```

2 раза
a = 6

```
a=4; b=6;
while(a<b) a+=b;
```

1 раз
a = 10

```
a=4; b=6;
while(a>b) a++;
```

0 раз
a = 4

```
a=4; b=6;
while(a<b) b=a-b;
```

1 раз
b = -2

```
a=4; b=6;
while(a<b) a--;
```

зацикливание

Замена `for` на `while` и наоборот

```
for( i=1; i<=10; i++)
   {
   // тело цикла
   }
```

```
i = 1;
while ( i <= 10 ) {
   // тело цикла
   i ++;
   }
```

```
for ( i=a; i>=b; i--)
   {
   // тело цикла
   }
```

```
i = a;
while ( i >= b ) {
   // тело цикла
   i --;
   }
```

! В языке Си замена цикла *for* на *while* и наоборот возможна всегда!

Задания

«4»: Ввести целое число и найти сумму его цифр.

Пример:

```
Введите целое число:
1234
Сумма цифр числа 1234 равна 10.
```

«5»: Ввести целое число и определить, верно ли, что в его записи есть две одинаковые цифры.

Пример:

```
Введите целое число:
1234
Нет.
```

```
Введите целое число:
1224
Да.
```

Последовательности

Примеры:

- 1, 2, 3, 4, 5, …
- 1, 2, 4, 7, 11, 16, …
- 1, 2, 4, 8, 16, 32, …
- $\frac{1}{2}, \frac{1}{2}, \frac{3}{8}, \frac{1}{4}, \frac{5}{32}, \ldots$

$a_n = n$

$a_1 = 1, \ a_{n+1} = a_n + 1$

$a_1 = 1, \ a_{n+1} = a_n + n$

$a_n = 2^{n-1}$

$a_1 = 1, \ a_{n+1} = 2a_n$

$\frac{1}{2}, \frac{2}{4}, \frac{3}{8}, \frac{4}{16}, \frac{5}{32}, \ldots$

$$a_n = \frac{b_n}{c_n}$$

$b_1 = 1, \ b_{n+1} = b_n + 1$

$c_1 = 2, \ c_{n+1} = 2c_n$

Последовательности

Задача: найти сумму всех элементов последовательности,

$$1,\ -\frac{1}{2},\ \frac{2}{4},\ -\frac{3}{8},\ \frac{4}{16},\ -\frac{5}{32},\ldots$$

которые по модулю больше 0,001:

$$S = 1 - \frac{1}{2} + \frac{2}{4} - \frac{3}{8} + \frac{4}{16} - \frac{5}{32} + \ldots$$

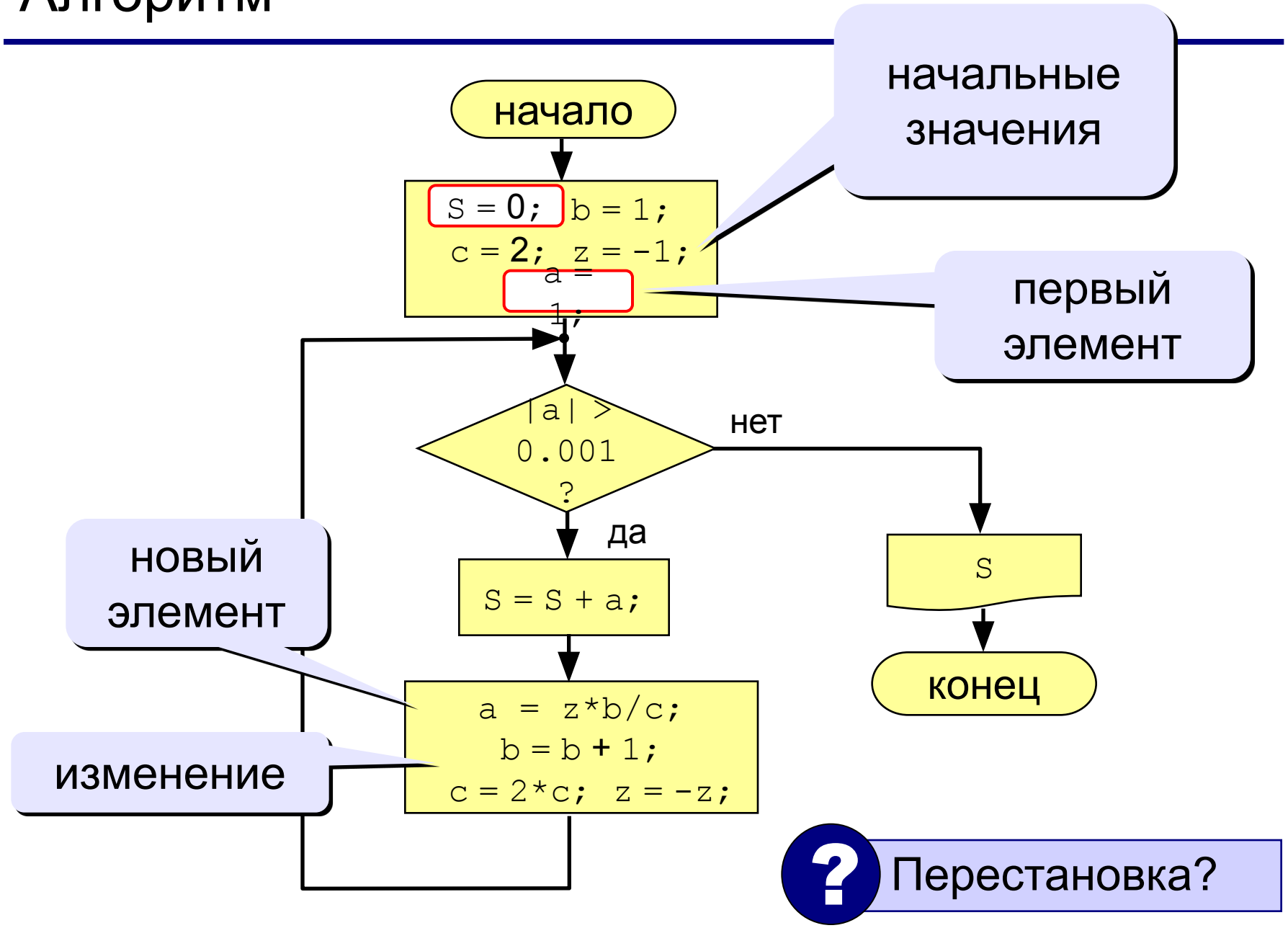
Элемент последовательности (начиная с №2):

$$a = z\,\frac{b}{c}$$

n	1	2	3	4	5	...
b	1	2	3	4	5	...
c	2	4	8	16	32	...
z	-1	1	-1	1	-1	...

```
b = b+1;
c = 2*c;
z = -z;
```

Алгоритм


начальные значения
начало
S = 0; b = 1;
c = 2; z = -1;
a = 1;
первый элемент
|a| > 0.001 ?
нет
да
S = S + a;
S
конец
новый элемент
a = z*b/c;
b = b + 1;
c = 2*c; z = -z;
изменение
Перестановка?

Программа

```
#include <math.h>
main()
{
  int b, c, z;
  float S, a, b;
  S = 0; z = -1;
  b = 1; c = 2; a = 1;
  while (fabs(a) > 0.001) {
    S += a;
    a = z * b / c;
    z = - z;
    b ++;
    c *= 2;
    }
  printf ("S = %10.3f", S);
}
```

математические функции

чтобы не было округления при делении

- модуль
ственного
числа

увеличение
суммы

расчет элемента последовательности

? Что плохо?

Задания

«4»: Найти сумму элементов последовательности с точностью 0,001:

$$S = 1 + \frac{2}{3 \cdot 3} - \frac{4}{5 \cdot 9} + \frac{6}{7 \cdot 27} - \frac{8}{9 \cdot 81} + \ldots$$

Ответ:

```
S = 1.157
```

«5»: Найти сумму элементов последовательности с точностью 0,001:

$$S = 1 + \frac{2}{2 \cdot 3} - \frac{4}{3 \cdot 9} + \frac{6}{5 \cdot 27} - \frac{8}{8 \cdot 81} + \frac{10}{13 \cdot 243} - \ldots$$

Ответ:

```
S = 1.220
```

Цикл с постусловием

Задача: Ввести целое положительное число (<2000000) и определить число цифр в нем.

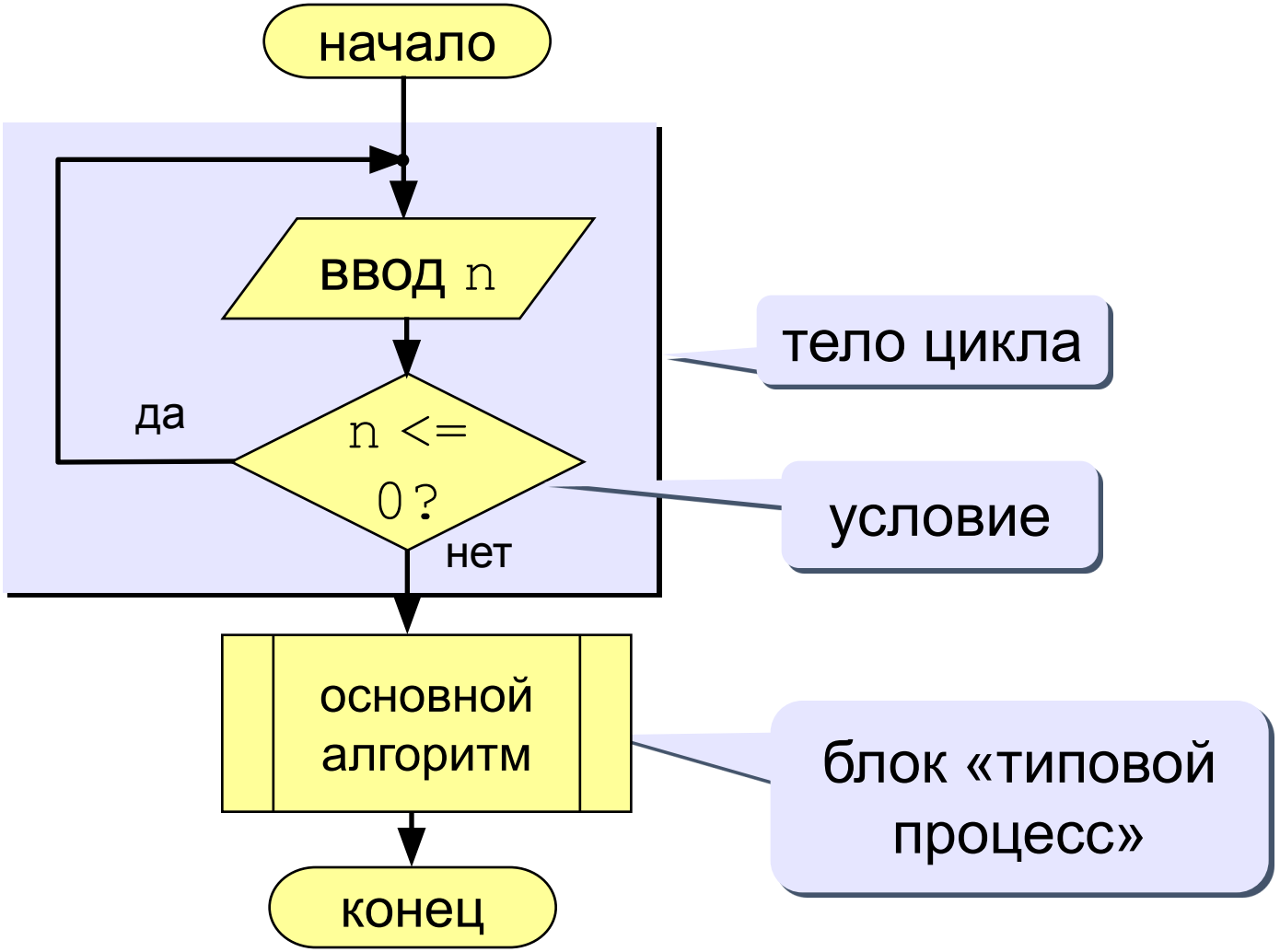
Проблема: Как не дать ввести отрицательное число или ноль?

Решение: Если вводится неверное число, вернуться назад к вводу данных (цикл!).

Особенность: Один раз тело цикла надо сделать в любом случае ⇒ проверку условия цикла надо делать в конце цикла (цикл с постусловием).

Цикл с постусловием – это цикл, в котором проверка условия выполняется в конце цикла.

Цикл с постусловием: алгоритм



Программа

```
main()
{
  int n;
  do {
    printf("Введите положительное число\n");
    scanf("%d", &n);
    }
  while ( n <= 0 );
  ... // основной алгоритм
}
```

условие

Особенности:

- тело цикла всегда выполняется хотя бы один раз
- после слова `while` («пока…») ставится условие продолжения цикла

Сколько раз выполняется цикл?

```
a = 4; b = 6;
do { a++; } while (a<=b);
```

3 раза
a = 7

```
a = 4; b = 6;
do { a += b; } while ( a<=b );
```

1 раз
a = 10

```
a = 4; b = 6;
do { a += b; } while ( a>=b );
```

зацикливание

```
a = 4; b = 6;
do b=a-b; while ( a>=b );
```

2 раза
b = 6

```
a = 4; b = 6;
do a+=2; while ( a>=b );
```

зацикливание

Задания (с защитой от неверного ввода)

«4»: Ввести натуральное число и определить, верно ли, что сумма его цифр равна 10.

Пример:

```
Введите число >= 0:
-234
Нужно положительное число.
Введите число >= 0:
1234
Да
```

```
Введите число >= 0:
1233
Нет
```

«5»: Ввести натуральное число и определить, какие цифры встречаются несколько раз.

Пример:

```
Введите число >= 0:
2323
Повторяются: 2, 3
```

```
Введите число >= 0:
1234
Нет повторов.
```

Программирование на языке Си

Тема 8. Оператор выбора

© К.Ю. Поляков, 2007-2008

Оператор выбора

Задача: Ввести номер месяца и вывести количество дней в этом месяце.

Решение: Число дней по месяцам:

28 дней – 2 (февраль)

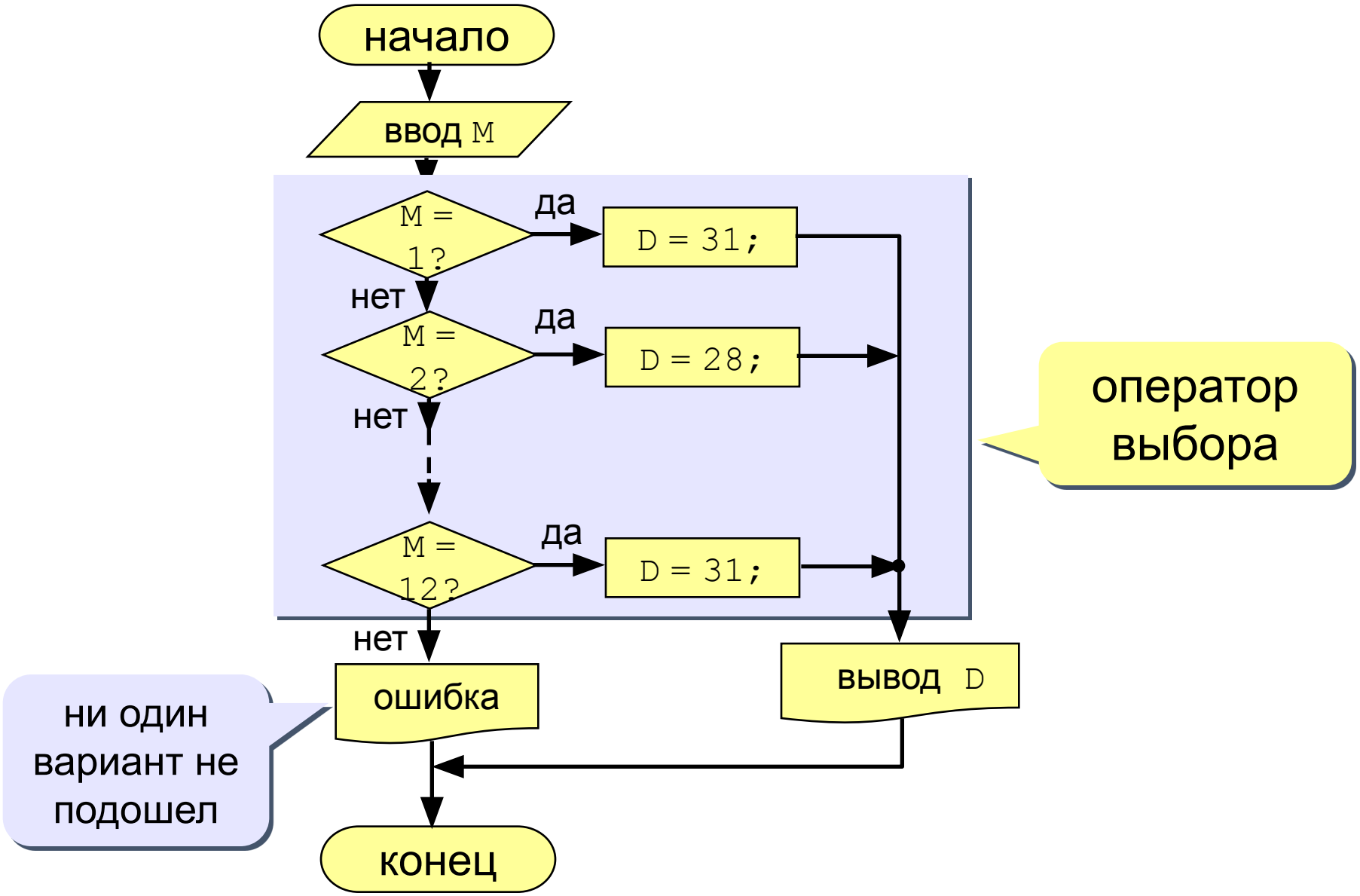
30 дней – 4 (апрель), 6 (июнь), 9 (сентябрь), 11 (ноябрь)

31 день – 1 (январь), 3 (март), 5 (май), 7 (июль), 8 (август), 10 (октябрь), 12 (декабрь)

Особенность: Выбор не из двух, а из нескольких вариантов в зависимости от номера месяца.

? Можно ли решить известными методами?

Алгоритм


начало
ввод M
M = 1?
да
D = 31;
нет
M = 2?
да
D = 28;
нет
M = 12?
да
D = 31;
нет
ошибка
вывод D
конец
оператор выбора
ни один вариант не подошел

Программа

```
main()
{
 int M, D;
 printf("Введите номер месяца:\n");
 scanf("%d", &M);
 switch ( M ) {
   case 2:  D = 28; break;
   case 4: case 6: case 9: case 11:
            D = 30; break;
   case 1: case 3: case 5: case 7:
   case 8: case 10: case 12:
            D = 31; break;
   default: D = -1;
   }
 if (D > 0)
      printf("В этом месяце %d дней.", D);
 else printf("Неверный номер месяца");
}
```

выйти из **switch**

ни один вариант не подошел

Оператор выбора

Задача: Ввести букву и вывести название животного на эту букву.

Особенность: выбор по символьной величине.

```
main()
{
 char c;
 printf("Введите первую букву названия животного:\n");
 scanf("%c", &c);
 switch ( c ) {
   case 'а': printf("Антилопа"); break;
   case 'б': printf("Бизон"); break;
   case 'в': printf("Волк"); break;
   default:  printf("Я не знаю!");
   }
}
```

? Что будет, если везде убрать `break`?

Оператор выбора

Особенности:

- после `switch` может быть имя переменной или арифметическое выражение целого типа (`int`)

```
switch ( i+3 ) {
  case 1: a = b; break;
  case 2: a = c;
  }
```

или символьного типа (`char`)

- нельзя ставить два одинаковых значения:

```
switch ( x ) {
  case 1: a = b; break;
  case 1: a = c;
  }
```

Задания (с защитой от неверного ввода)

«4»: Ввести номер месяца и вывести количество дней в нем, а также число ошибок при вводе.

Пример:

```
Введите номер месяца:
-2
Введите номер месяца:
11
В этом месяце 30 дней.
Вы вводили неверно 1 раз.
```

```
Введите номер месяца:
2
В этом месяце 28 дней.
Вы вводили неверно 0 раз.
```

«5»: Ввести номер месяца и номер дня, вывести число дней, оставшихся до Нового года.

Пример:

```
Введите номер месяца:
12
Введите день:
25
До Нового года осталось 6 дней.
```

Программирование на языке Си

Тема 9. Отладка программ

© К.Ю. Поляков, 2007-2008

Отладка программ

Отладка – поиск и исправление ошибок в программе.
Англ. *debugging*, *bug* = *моль, жучок*

Методы:

- трассировка – вывод сигнальных сообщений
- отключение части кода (в комментарии)
- пошаговое выполнение – выполнить одну строчку программы и остановиться
- точки останова – выполнение программы останавливается при достижении отмеченных строк (переход в пошаговый режим)
- просмотр и изменение значений переменных в пошаговом режиме

Трассировка

```
main()
{
 int i, X;
 printf("Введите целое число:\n");
 scanf("%d", &X);
 printf("Введено X=%d\n", X);
 for(i=1; i<10; i++)
    {
    printf("В цикле: i=%d, X=%d\n", i, X);
    ...
    }
 printf("После цикла: X=%d\n", X);
 ...
}
```

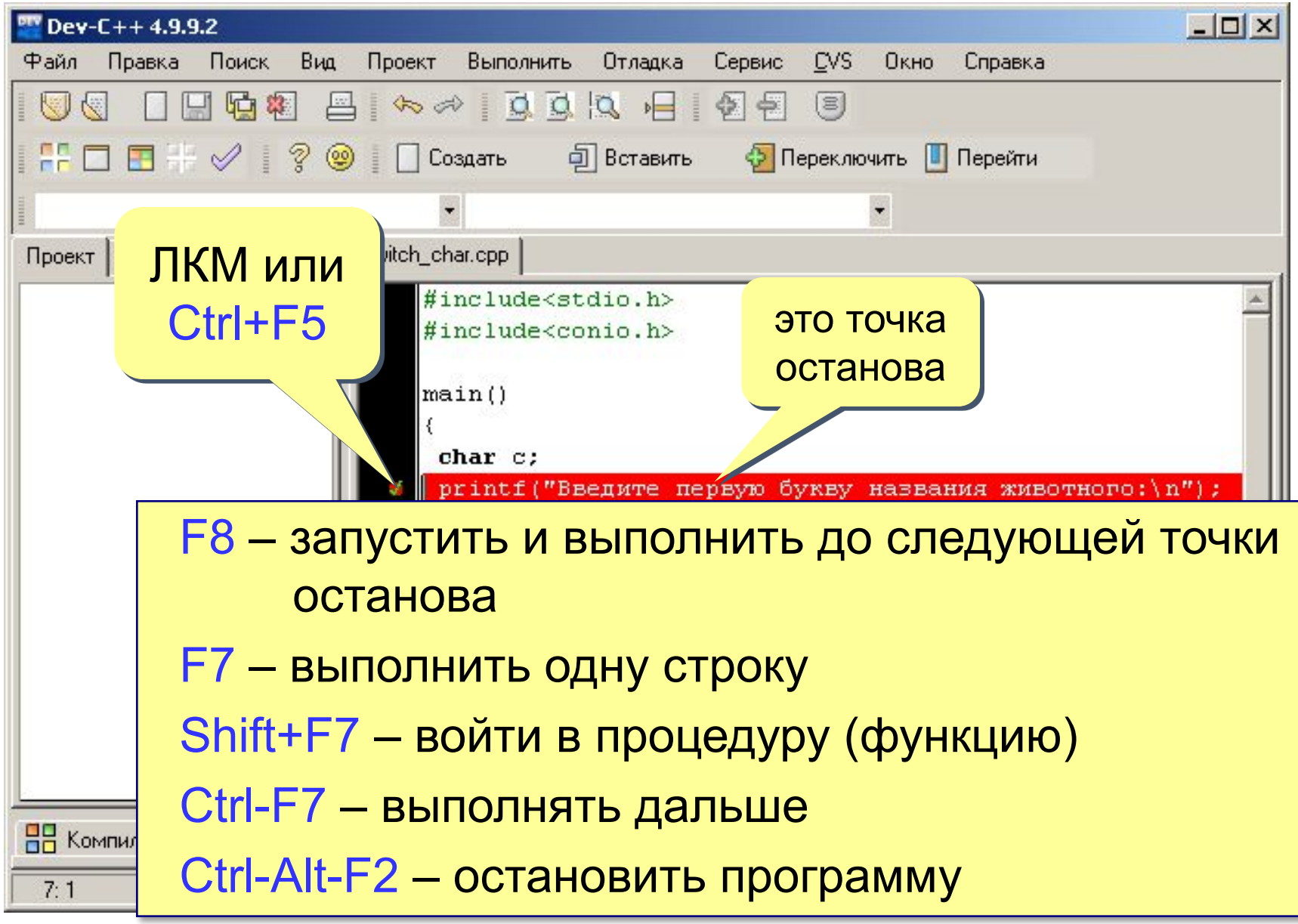
Отключение части кода (комментарии)

```
main()
{
 int i, X;
 printf("Введит целое число:\n");
 scanf("%d", &X);
 // X *= X + 2;
 for(i=1; i<10; i++) X *= i;
 /* while ( X > 5 ) {
    ...
    } */
 ...
}
```

комментарий до конца строки //

закомментированный блок /* … */

Точки останова



F8 – запустить и выполнить до следующей точки останова

F7 – выполнить одну строку

Shift+F7 – войти в процедуру (функцию)

Ctrl-F7 – выполнять дальше

Ctrl-Alt-F2 – остановить программу

Просмотр значений переменных

Dev-C++ 4.9.9.2

Файл Правка Поиск Вид Проект Выполнить Отладка Сервис CVS Окно Справка

Создать Вставить Переключить Перейти

Проект Классы Отладка

sum.cpp

```
#include<stdio
#include<conio
main()
{
   int a, b,
```

ПКМ

навести мышь на имя переменной

Добавить в наблюдаемые F4

Изменить значение

Удалить объект наблюдения

Clear all

Отладка Результаты поиска Закрыть

Отладка Отслеживать Вывод

Следующий шаг | Продолжить выполнение | Отладка | Добавить в наблюдаемые

Шаг внутрь | Выполнить до курсора | Остановить выполнение | Удалить объект наблюдени

8: 12 Вставка 12 строк в файле