Программирование на языке Java

3. Алгоритмы и программы
4. Знакомство с Знакомство с Java
5. Переменные

Программирование на языке Java

Тема 3. Алгоритмы и программы

Программирование

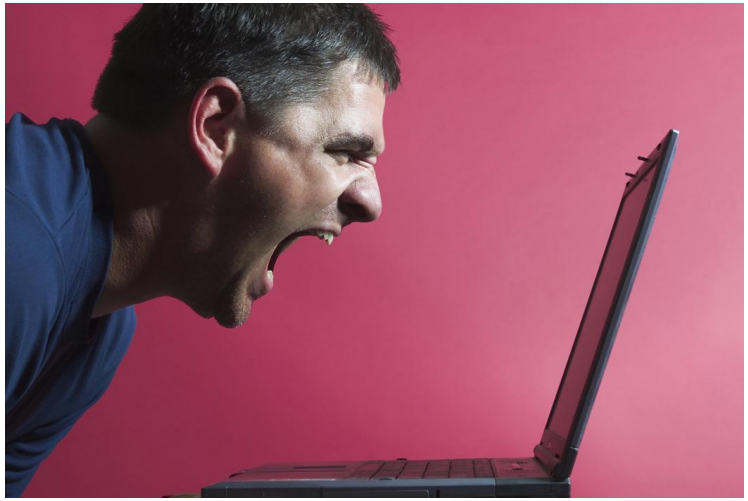
«Любой человек должен:
уметь сменить пеленку,
составить план вторжения,
заколоть свинью,
вести корабль,
построить дом,
написать сонет,
подвести счета,
построить стену,
снять мясо с костей,
утешить умирающего,
отдать приказ,
выполнить приказ,
действовать вместе и в одиночку,
решать уравнения,
анализировать новую проблему,
разбросать навоз,
запрограммировать компьютер,
приготовить вкусное блюдо,
биться и победить,
и умирать с достоинством.

Специализированны лишь насекомые.»

Р. Хайнлайн.

Достаточно времени для любви, или жизни Лазаруса Лонга, 1973

Программирование



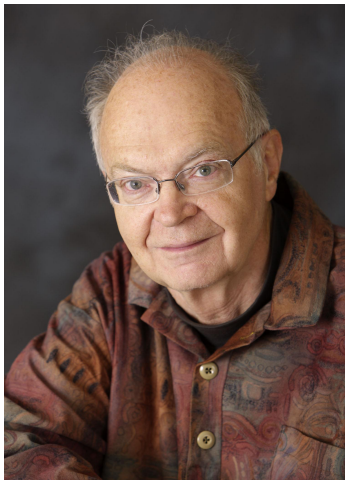
Скажи компьютеру что делать

Программирование:

- не только для экспертов;
- естественный, приносящий удовольствие творческий опыт;
- приносит новые достижения;
- путь в новый мир интеллектуальных соревнований.

Вызовы программирования

- Нужно узнать, на что способны компьютеры.
- Необходимо выучить язык программирования



Дональд Кнут

Вместо того, чтобы представлять себе, что наша главная задача – обучить компьютер тому, что делать, давайте сосредоточимся на объяснении людям того, что мы хотим, чтобы компьютер делал.

Алгоритм

Алгоритм – точный набор инструкций, описывающий порядок действий исполнителя для достижения результата решения задачи за конечное время.

Свойства алгоритма

- **дискретность**: состоит из отдельных шагов (команд)
- **понятность**: должен включать только команды, известные исполнителю (входящие в СКИ)
- **детерминированность (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
- **результативность**: завершает работу определёнными результатами

Задача

Является ли формула алгоритмом для вычисления числа π

$$\pi = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots\right) = 4\sum_{i=0}^{\infty}\frac{(-1)^i}{2i+1}$$

Не соблюдаются свойства:

- **конечности**
- **массовости**

Программа

Программа – запись алгоритма на формальном языке (часто употребляется как синоним термина алгоритм).

Алгоритм – основная идея, метод, схема решения задачи.

Программа – конкретная реализация алгоритма, которая может быть скомпилирована и выполнена на компьютере.

Команды – 1

Команда – описание действий, которые должен выполнить исполнитель.

- откуда взять исходные данные?
- что нужно с ними сделать?

Исполнитель должен уметь выполнять все команды, составляющие алгоритм.

Множество возможных команд **конечно** и изначально **строго задано**.

Действия, выполняемые по этим командам, называются **элементарными**.

Команды – 2

У каждого исполнителя есть конечный набор элементарных команд (действий), оперирующих элементарными объектами, которых также конечное число.

Способы записи алгоритма. Словесный – 1

Словесная запись – описание последовательных этапов обработки данных в произвольном изложении на естественном языке.

Недостатки:

- отсутствие строгой формализации;
- многословность записи;
- допускают неоднозначность толкования отдельных предписаний.

Способы записи алгоритма. Словесный – 2

Пример.

1. задать два числа;
2. если числа равны, то взять любое из них в качестве ответа и остановиться, в противном случае продолжить выполнение алгоритма;
3. определить большее из чисел;
4. заменить большее из чисел разностью большего и меньшего из чисел;
5. повторить алгоритм с шага 2.

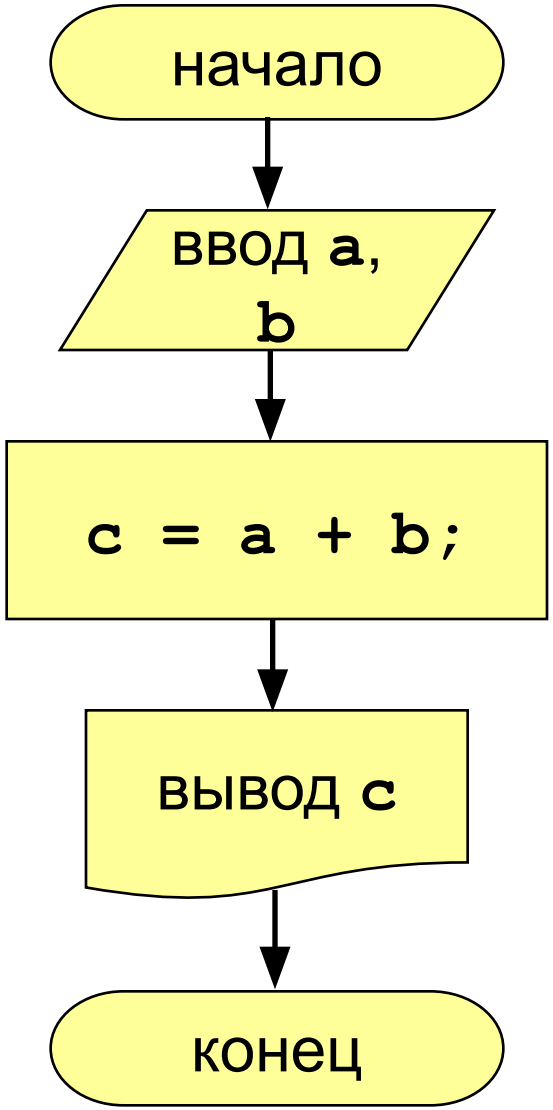
Способы записи алгоритма. Графический – 1

Графическое представление – алгоритм изображается в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий.

Такое графическое представление называется **блок-схемой**.

Способы записи алгоритма. Графический – 2

Пример.



Способы записи алгоритма. Псевдокод – 1

Псевдокод – полуформализованное описание алгоритмов на условном алгоритмическом языке, включающее в себя как элементы ЯП, так и фразы естественного языка, общепринятые математические обозначения и др.

Псевдокод – промежуточный язык между естественными и формальными языками.

Способы записи алгоритма. Псевдокод – 2

Пример.

```
алг HELLOWORLD
нач
вывод ('Hello, World')
кон алг HELLOWORLD
```

Способы записи алгоритма. Программа

Программа – запись на языке программирования.

Пример программы на Java

```
public class First {
public static void main (String[] args)
   {
   System.out.print("Hello, World!");
   }
}
```

Основные качества программ

- **Корректность (правильность)** – реализация корректного алгоритма решения исходной задачи.
- **Эффективность** – уменьшение времени работы программы.
- **Понятность и модифицируемость**
- **Удобство эксплуатации**
- **Надежность**
- **Удобство сопровождения**

Писать понятные и легко модифицируемые программы существенно легче, чем правильные и эффективные.

Правила написания программного кода

1. Использовать осмысленные имена для всех переменных, отличных от счетчиков;
2. Константам, отличным от нуля и единицы присваивать имена;
3. Соблюдать принятый в языке стиль написания имен (имена классов с прописной буквы, переменных и методов – со строчной, констант – полностью из прописных);
4. Добавлять краткие и понятные комментарии.
5. Применять форматирование текста (лесенка – упорядочивание программного кода в целях повышения его читабельн

В NetBeans автоматическое форматирование
Alt + Shift + F

Этапы разработки программ – 1

1. Постановка задачи
2. Анализ и исследование задачи, модели
3. Разработка алгоритма
4. Программирование
5. Тестирование и отладка
6. Анализ результатов решения задачи
7. Сопровождение программы

Этапы разработки программ – 2

1. Постановка задачи:

- сбор информации о задаче;
- формулировка условия задачи;
- определение конечных целей решения задачи;
- определение формы выдачи результатов;
- описание данных (их типов, диапазонов величин, структуры и т.п.).

Этапы разработки программ – 3

2. Анализ и исследование задачи, модели:

- анализ существующих аналогов;
- анализ технических и программных средств;
- разработка математической модели;
- разработка структур данных.

Этапы разработки программ – 4

3. Разработка алгоритма:

- выбор метода проектирования алгоритма;
- выбор формы записи алгоритма (блок-схемы, псевдокод и др.);
- выбор тестов и метода тестирования;
- проектирование алгоритма.

Этапы разработки программ – 5

4. Программирование:

- выбор языка программирования;
- уточнение способов организации данных;
- запись алгоритма на выбранном языке программирования.

Этапы разработки программ – 6

5. Тестирование и отладка:

- синтаксическая отладка;
- отладка семантики и логической структуры;
- тестовые расчеты и анализ результатов тестирования;
- совершенствование программы.

Этапы разработки программ – 7

6. Анализ результатов решения задачи:

- уточнение в случае необходимости математической модели с повторным выполнением этапов 2 — 5

7. Сопровождение программы:

- доработка программы для решения конкретных задач;
- составление документации к решенной задаче, к математической модели, к алгоритму, к программе, к набору тестов, к использованию.

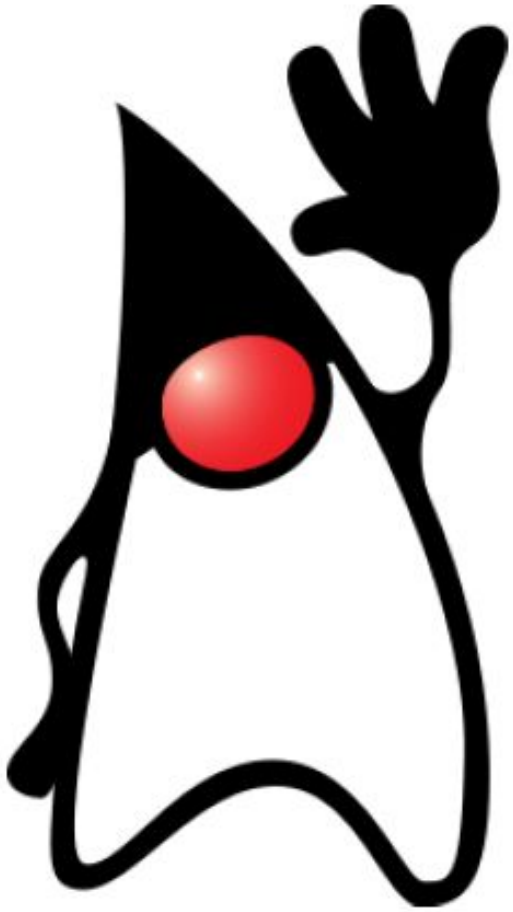
Программирование на языке Java

Тема 4. Знакомство с Java

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*: Basic, ЛОГО, Pascal
  - *профессиональные*: Java, C, C++
  - *для задач искусственного интеллекта*: Prolog, LISP
  - *для Интернета*: JavaScript, Java, Perl, PHP, ASP.Net, Ruby


Java™



Язык Java

Java – объектно-ориентированный язык программирования, разработанный Sun Microsystems в 1995 г.

Почему стоит изучать Java?

Один из самых популярных и востребованных ЯП.
Индекс TIOBE (https://www.tiobe.com/tiobe-index/)
Рейтинг CFF языков программирования (https://m.habr.com/company/it-grad/blog/422057/)

На Java пишут:

- высоконагруженные системы;
- корпоративные приложения;
- настольные приложения;
- программы и игры для телефонов, в том числе под Android
- апплеты для смарт-карт

- Язык развивается и совершенствуется (версия Java 12 вышла в марте 2019)

Почему стоит изучать Java?

Java — это не только язык программирования, но и. . .

- обширная стандартная библиотека;
- сторонние библиотеки и фреймворки;
- инструменты разработки (сборка, тестирование);
- методология ООП, паттерны проектирования;
- платформа для альтернативных языков; (Clojure, Groovy, JRuby, Jython, Kotlin, Scala).

Виртуальная машина и байт-код – 1

Традиционный подход:

исходный код → машинный код → процессор

- программа работает только на той платформе, под которую она скомпилирована

Подход Java:

исходный код → байт-код виртуальной машины → виртуальная машина → процессор

- программа работает на любой платформе, где есть виртуальная машина Java
- **“Write once, run anywhere!”** («Написано однажды, работает везде!»)

Виртуальная машина и байт-код – 2

Программы транслируются в байт-код, выполняемый виртуальной машиной Java (JVM = Java Virtual Machine).

Байт-код – машинно-независимый код низкого уровня, генерируемый транслятором и исполняемый виртуальной машиной.

Большинство инструкций байт-кода эквивалентны одной или нескольким командам ассемблера.

Виртуальная машина и байт-код – 3

Виртуальная машина Java (JVM) – основная часть исполняющей системы Java, интерпретирует и исполняет байт-код Java.

JVM доступны для многих аппаратных и программных платформ, что обеспечивает **кросс-платформенность** Java.

Виртуальная машина и байт-код – 4

Насколько быстро работает виртуальная машина?

- Интерпретация байткода на порядок (10–20 раз) медленнее исполнения аналогичного машинного кода. . .

- но есть Just-In-Time компиляция – виртуальная машина компилирует байткод в машинный код (используется с JDK 1.1)
- а также HotSpot адаптивный оптимизирующий JIT-компилятор (используется с JDK 1.3)
- в результате Java 8 всего в 1.5–2 раза медленнее C, а в некоторых тестах не хуже или даже быстрее!

Сборка мусора

Подход C/C++:

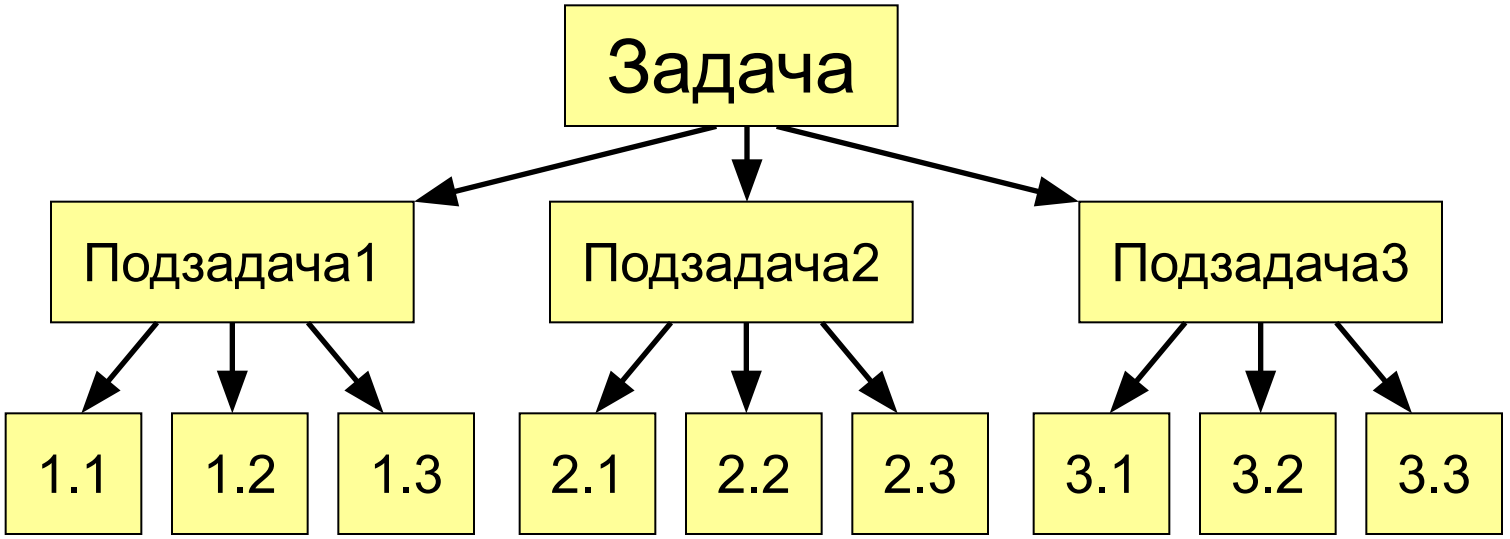
- выделил память → поработал → освободил память
- всё управление памятью в руках программиста

Подход Java:

- выделил память → поработал → молодец
- виртуальная машина считает ссылки на каждый объект
- освобождает память, когда ссылок больше нет

Разработка программ на Java

- разработка программ «сверху вниз»



- разнообразные структуры данных (массивы, коллекции: списки, отображения, множества)
- объектно-ориентированный подход


HELLO
WORLD

Из чего состоит программа?

```
public class <имя класса>
   {
   public static void main(String[]
args)
         {
      /* основная программа */
      }
   }
```

Комментарии, заключенные в «скобки» /* */ не обрабатываются

Простейшая программа

имя класса должно совпадать с именем файла

void = «пустой» основной метод не возвращает никакого результата

главный (основной) метод класса всегда имеет имя ***main***

```
public class <имя класса>
      {
      public static void main(String[] args)
            {
            /* основная программа */
            }
      }
```

«тело» метода (основная часть)

начало метода

конец метода

Что делает эта программа?

Что происходит дальше?

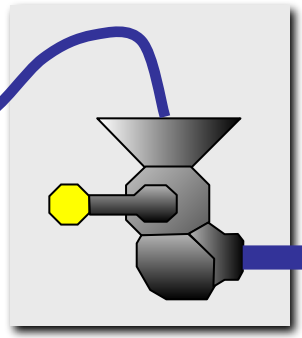
текст программы на Java

First.java

```
public class
{
...
}
```

исходный файл

транслятор



First.class

```
ЪБzЦ2?|ё3БКа
n/36ШпIC+И -
ЦЗ_5 МуРЧ б
s6bд^:/@:лЖ1_
```

байт-код

!

- **по исходному файлу можно восстановить байт-код**
- **байт-код можно запустить**

Вывод текста на экран

стандартные функции вывода

```
public class HelloWorld
{
   public static void main (String[] args)
      {
      System.out.print("Привет!");
      }
}
```

вызов стандартного метода ***print*** (вывод)

этот текст будет выведен на экран

Переход на новую строку

последовательность \n
код 10
переход на новую строку

```
public class HelloWorld
{
public static void main (String[] args)
   {
   System.out.print(" Привет,\nВася!");
   }
}
```

на экране:

```
Привет,
Вася!
```

Задания

1. Вывести на экран текст "лесенкой"

```
Вася
    пошел
        гулять
```

2. Вывести на экран рисунок из букв

```
  Ж
 ЖЖЖ
ЖЖЖЖЖ
ЖЖЖЖЖЖЖ
НН  НН
ZZZZZ
```

Программирование на языке Java

Тема 5. Переменные

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 в интервале $[-2^{31}, 2^{31}-1]$ (4 байта)
- **float** – вещественное число, *floating point* (4 байта)
- **char** – символ, *character* (2 байта)

Из чего состоит программа?

Переменная – изменяющаяся величина, имеющая имя (ячейка памяти).

Метод – вспомогательный алгоритм, описывающий некоторые действия (например, рисование окружности) или выполняющий вычисления (например, вычисление квадратного корня, **sin**).

Имена классов, переменных, методов

В Java имена могут включать

- Символы алфавита (латиница A-Z, кириллица А-Я и т.д.)

заглавные и строчные буквы различаются

- цифры

имя не может начинаться с цифры

- знак подчеркивания _, знак $

Имена НЕ могут включать

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

Какие имена правильные??

AXby R&B 4Wheel Вася “PesBarbos”
TU154 [QuQu] _ABBA A+B

Ключевые слова Java – 1

Ключевые слова в сочетании с синтаксисом операций и разделителями образуют основу языка Java.

Ключевые слова **нельзя** использовать в качестве имен переменных, классов, методов.

Ключевые слова Java – 2

`abstract`	`continue`	`for`	`new`	`switch`
`assert`	`default`	`goto`	`package`	`synchronized`
`boolean`	`do`	`if`	`private`	`this`
`break`	`double`	`implements`	`protected`	`throw`
`byte`	`else`	`import`	`public`	`throws`
`case`	`enum`	`instanceof`	`return`	`transient`
`catch`	`extends`	`int`	`short`	`try`
`char`	`final`	`interface`	`static`	`void`
`class`	`finally`	`long`	`strictfp`	`volatile`
`const`	`float`	`native`	`super`	`while`

Ключевые слова `const` и `goto` зарезервированы, но не используются.

Зарезервированные слова Java

`true, false, null` – зарезервированные слова в Java, нельзя использовать в качестве имен переменных, классов и т.п.

Объявление переменных

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

```
public static v
{
i
f
int Tu104, Il86=23, Yak42;
float x=4.56, y, z;
char c, c2='A', m;
}
```

Целочисленная 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'

Переменные

Переменная – это величина, имеющая имя, тип и значение. Значение переменной можно изменять во время работы программы.

Объявление переменных (выделение памяти):

```
int a, b;
double Q;
char s1, s2;
```

Простые типы данных (primitive)

char { символьный (один символ) }

можно использовать русские буквы!

byte, short, int, long {целые типы}

float, double { вещественные типы }

целая и дробная часть отделяются точкой

boolean { логический }

может принимать два значения:

- true (истина, «да»)
- false (ложь, «нет»)

Как изменить значение переменной?

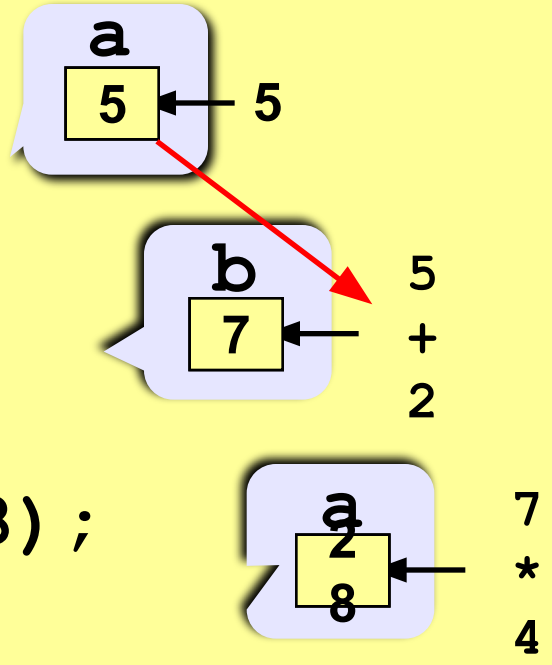
Оператор – это команда языка программирования высокого уровня.

Оператор присваивания служит для изменения значения переменной.

Пример:

```
{
int a, b;

   a = 5;
   b = a + 2;
   a = (a + 2)*(b - 3);
 }
```



Оператор присваивания

куда

что

```
<имя переменной> = <выражение>;
```

Арифметическое выражение может включать

- имена переменных
- знаки арифметических операций:

```
+  -    *    /    %
```

умножение

деление

остаток от деления

- вызовы методов
- круглые скобки ( )

Для чего служат круглые скобки?

Какие операторы записаны неверно?

```
public static void main (String[] args)
{
 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**

нельзя записывать вещественное значение в целочисленную переменную

Особенность деления в Java

При делении целых чисел остаток отбрасывается!

```
public static void main(…)
{
int a = 7;
float x;
x = a / 4;
x = 4 / a;
x = (float)a / 4;
x = 1.*a / 4;
}
```

1

0

1.75

1.75

Сокращенная запись операций в Java

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

```
public static void
 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
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
x =(a*a+5*c*c-d*(a+b))/((c+d)*(d-2*a));
```

Сложение двух чисел

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

Простейшее решение:

```
import java.util.Scanner;
public static void main (…) {
Scanner in = new Scanner (System.in);
 int a, b, c;
 System.out.print("Введите a");
 a = in.nextInt();
 System.out.print("Введите b");
 b = in.nextInt();
 c = a + b;
 System.out.println(c);
 }
```

Подключение пакета

Создание экземпляра класса

Чтение из входного потока целого числа

Ввода данных с клавиатуры

```
Scanner in = new Scanner (System.in);
a = in.nextInt ();     /* ввод целого
 значения и присваивание переменной a */

b = in.nextDouble (); /* ввод
 вещественного значения и присваивание
 переменной b */
```

Оператор вывода

```
System.out.print ( a );    /* вывод
                  значения переменной a */

System.out.println ( a ); /* вывод
                  значения переменной a
                  и переход на новую
                  строчку /*
System.out.println ( “Привет!” ); /*
                  вывод текста */
System.out.println( “Ответ: ” + c );
          /* вывод текста и значения
          переменной c */
System.out.println ( a + “+” + b+ “=”
          + c );
```

Полное решение

```
 int a, b, c;
System.out.println("Введите число a");
 a = in.nextInt();
System.out.println("Введите число b");
 b = in.nextInt();
 c = a + b;
 System.out.println(a+ "+                c );
```

компьютер

Протокол:

Введите число a

пользователь

25

Введите число b

компьютер

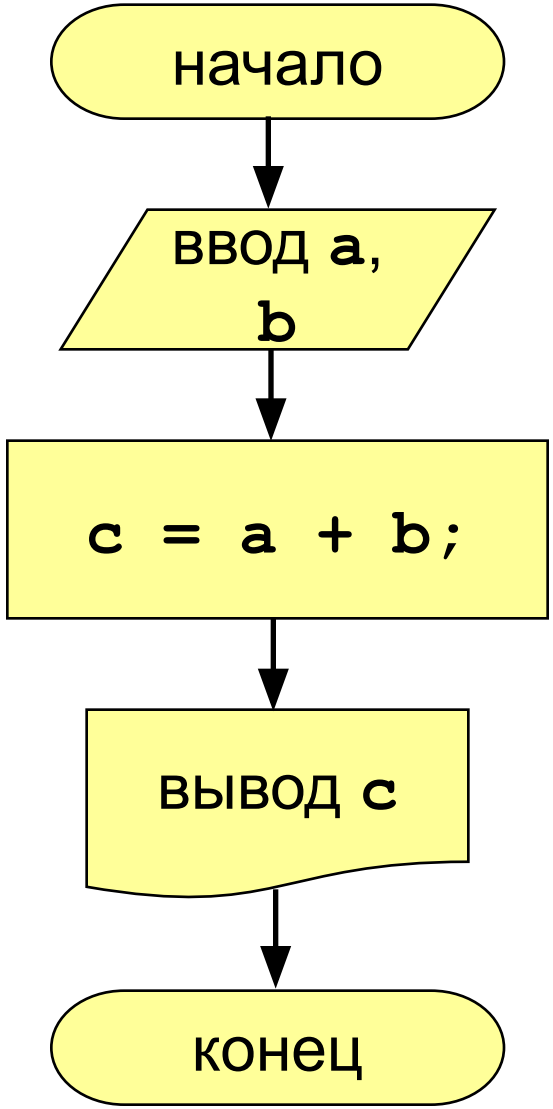
30

пользователь

25+30=55

компьютер

Блок-схема линейного алгоритма



блок «начало»

блок «ввод»

блок «процесс»

блок «вывод»

блок «конец»

Задания

1. **Ввести три числа, найти их сумму и произведение.**

Пример:

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

2. **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
```