

Конспект лекций по java

Александр Титилин

Содержание

1	Шапочка	3
2	Hello World	3
3	Целочисленный тип данных int	3
4	Ввод с клавиатуры	4
4.1	Получение целого числа с клавиатуры.	4
5	Действия с целыми числами	4
5.1	Замечание. Деление отрицательных чисел	4
6	Целочисленные типы данных в java	5
7	Ерунда про двоичное представление, про знак и прочее	5
8	15.09.2022	5
8.1	Вещественный тип данных	5
8.2	Ввод	5
8.3	Вывод на экран	5
8.4	Операции с вещественными числами	5
8.5	Приведения действительных чисел к целым	6
9	Математические функции в java.	6
9.1	Символьный тип данных char	6
9.2	Ввод с клавиатуры	6
10		6
10.1	Логический тип данных	6
10.2	Операции сравнения	7
10.3	Логические операции	7
10.4	Условный оператор	7
10.5	Область видимости переменных.	8

11 Цикл с счетчиком <u>for</u>	8
11.1 Задачака	8
11.1.1 Первый способ, формулкой	8
11.1.2 Второй способ, дополнительная переменная	8
11.2 Задача	9
11.3 Правило	9
12 16.10.2022	9
12.1 Поиск максимума.	9
12.2 Цикл с условием	10
12.2.1 Цикл с предусловием.	10
12.2.2 Цикл с предусловием	10
12.3 Примеры	10
12.3.1 Последний элемент не нужно учитывать.	11
12.3.2 Последний элемент нужно учитывать	11
12.3.3 Еще пример	11
13 Массивы.	12
13.1 Описание и создание массива	12
13.1.1 Описание	12
13.1.2 Создание	12
13.1.3 Прошлые действия в одном	12
13.2 Заполнение массива.	12
13.3 Вывод массива на экран.	13
13.4 Подсчет элементов	13
14 Перестановки	13
15 Поиск минимума и максимума.	13
16 Заполнение массива случайными числами	14
17 20.10.20	14
17.1 Однопроходные алгоритмы.	14
17.2 Обработка строк	14
17.2.1 Пример	15
18 27.10.22	15
18.1 Подпрограммы(статические методы)	15
18.2 Подпрограммы без параметров	15
18.2.1 Вызов подпрограммы.	16
18.2.2 Примеры	16
18.3 Подпрограммы с параметрами.	16
18.3.1 Пример	16

19 03.11.2022	17
19.1 Рекурсия.	17
19.2 Структура рекуррентной программы	17
20 10.11.2022	18
20.1 Объектно-ориентированное программирование.	18

1 Шапочка

```
public class Main {
    public static void main(String[] args){
        write here
    }
}
```

2 Hello World

```
System.out.print("Hello World");
System.out.println();
System.out.print("How are you?");
```

print - вывести в консоль без переноса строки.
println - вывести в консоль с переносом строки.

3 Целочисленный тип данных int

Создание переменной с именем variableName с типом dataType

```
dataType variableName;
```

Переменная sum с типом данных int;

```
int sum;
```

Положим в sum число 12.

```
sum = 12;
```

Создаем переменную sum и сразу присваиваем ей значение 12

```
int sum = 12;
```

Описываем несколько переменных и некоторые инициализируем

```
int a , b , c = 5, d , e = 1234;
```

Вывод значение переменной sum на экран.

```
System.out.println(sum);
```

Выводить можем и значения выражений.

```
System.out.println(sum + 129291);
```

```
System.out.println("next after sum" + sum + 1);
```

Напишет "... 121"

```
System.out.println("next after sum" + ( sum + 1 ));
```

Напишет "... 13"

int - целое число, размером 4 байта. Имена переменных начинаются с маленькой буквы, состоящие из нескольких слов используют camelCase (countOfEvenDigits).

4 Ввод с клавиатуры

Для ввода с клавиатуры нужен объект Scanner из шапки с сайта школы.

4.1 Получение целого числа с клавиатуры.

```
sum = in.nextInt();
```

Так тоже можно

```
int second = in.nextInt();
```

5 Действия с целыми числами

Арифметические действия как везде (*, +, -, /, %), деление целочисленное. Побитовый сдвиг влево

```
a << b;
```

Побитовый сдвиг вправо.

```
a >> b;
```

Знаковый побитовый сдвиг вправо.

```
a >>> b;
```

5.1 Замечание. Деление отрицательных чисел

Делимое	Делитель	Целое	Остаток
23	5	4	3
-23	5	-4	-3
23	-5	4	3
-23	-5	4	-3

В питоне и математике не так.

6 Целочисленные типы данных в java

Тип Данных	Размер ячейки	Размер в битах	Диапазон
byte	1 байт	8	$-128 \dots 127$
short	2 байта	16	$-32768 \dots + 32767$
int	4 байта	32	$-2^{31} + \dots 2^{31} - 1$
long	8 байт	64	$-2^{63} + \dots + 2^{63} - 1$

7 Ерунда про двоичное представление, про знак и прочее

Суть в том, что старший разряд в двоичном представлении это знак. Про дополнительный двоичный код. Пока джавы нет идут байки про работу компа.

8 15.09.2022

8.1 Вещественный тип данных

Вещественные типы данных в java - float(4 байта) и double(8 байт). Если можно не использовать вещественные числа, то их не надо использовать. Прикол про $0.1 + 0.2 \neq 0.3$. Любые числа с вещественными числами будут приближенными.

```
float f = 1.7;  
double d = 1.7;
```

Первое работать не будет.

8.2 Ввод

```
double x = in.nextDouble() ;
```

8.3 Вывод на экран

```
out.printf("%.3f",x);
```

Вывод вещественного числа x с 3 числами после запятой.

```
out.printf("Answer: %d %.2f\n",a,x);
```

Вывод слова, целого числа a в десятичном представлении и вещественного числа x с двумя числами после запятой и перевод строки.

8.4 Операции с вещественными числами

Арифметические как в целых, кроме деления.

```
int b = 23, c=5;  
double y = b/c;
```

Y будет равен четырем. Надо делать так

```
double y = (double)b / c;
```

Надо какое нибудь число привести к вещественным.

8.5 Приведения действительных чисел к целым

```
int g = (int) x;
```

9 Математические функции в java.

Все такие функции лежат в библиотеке Math, подключать не надо.

Math.abs(x)	$ x $
Math.sqrt(x)	\sqrt{x}
Math.sin(x)	$\sin x$
Math.cos(x)	$\cos x$
Math.tan(x)	$\tan x$

9.1 Символьный тип данных char

```
char c = 'F';
```

Представляет собой целое беззнаковое число, занимает 2 байта.

9.2 Ввод с клавиатуры

```
char h = (char) System.in.read();
```

При этом компилятор ругнется. Нужно использовать альтернативную шапочку

```
import java.io.IOException
public class Main {
    public static void main(String[] args) throws IOException
    {
        write here
    }
}
```

10

10.1 Логический тип данных

```
boolean b = true;
```

10.2 Операции сравнения

Математика	Java
>	>
<	<
=	==
≥	>=
≤	<=
≠	!=

```
boolean c = 3 > 5;  
out.print(c);
```

Считывать boolean нельзя.

10.3 Логические операции

Операция	Обозначение	Смысл
Не (инверсия)	!	Меняет логическое значение на противоположное
И (конъюнкция)	&&	Истина, если оба операнда истина
Или (дизъюнкция)		Истина, если хотя бы один операнд истина
Xor	^	Истинна если операнды разные

В java нельзя использовать двойные сравнения.

10.4 Условный оператор

```
if (cond){  
    operator-Yes;  
}
```

```
if (cond){  
    operator-Yes;  
}  
else {  
    operator-No;  
}
```

Примеры.

```
if (x > 10) {  
    System.out.println("Too much");  
}
```

```
if (x > 10){  
    System.out.println("Too much");  
}  
else {  
    System.out.println("Good");  
}
```

10.5 Область видимости переменных.

Создание переменной внутри блока

```
{  
    int a = 10;  
}
```

Внутри скобок использовать можно, снаружи нет.

11 Цикл с счетчиком for

Цикл повторяющаяся последовательность действий, которые называются телом цикла.

```
for(start values; condition;change counter) {  
    body  
}
```

В блоке начальных значений, можно описывать переменные. В блоке изменения счетчика можно менять несколько начальных значений.

```
for(int i = 0,k,j = ; i < 10 && j < 1000;i++,j+10)
```

Эти переменные пропадут, после окончания цикла.

```
for(int i = 5; i <= 8; i ++)
```

Выведет

```
5  
6  
7  
8
```

11.1 Задача

Дано натуральное число, нужно вывести первые n четных чисел.

11.1.1 Первый способ, формулой

```
for(int i = 0; i < n ; i++){  
    out.println(2*(i + 1));  
}
```

11.1.2 Второй способ, дополнительная переменная

```
int a = 2  
for(int i = 0; i < n; i++){  
    out.println(a);  
    a += 2;  
}
```


Тоже самое, но короче

```
for (int i = 0, a = 2; i < n; i++, a+=2) {  
    out.println(a);  
}
```

11.2 Задача

Дано число n (количество элементов последовательности), после этого даны n чисел. Надо найти \sum четных элементов данной последовательности

```
int n = in.nextInt();  
int sum = 0;  
for (int i = 0, a; i < n ; i++){  
    a = in.nextInt();  
    if (a % 2 == 0){  
        sum +=a;  
    }  
}  
out.println(sum);
```

11.3 Правило

Начальные значения переменных (суммы, количества) нужно задавать непосредственно перед тем циклом, в котором они изменяются.

12 16.10.2022

12.1 Поиск максимума.

Задача 1. Дано число n . Затем ищем еще n целых чисел. Найти максимальный элемент.

```
int n = in.nextInt();  
int max = in.nextInt();  
int x;  
for (int i = 0; i < n - 1; i++){  
    x = in.nextInt();  
    if (x > max){  
        max = x;  
    }  
}  
out.println(max);
```

Задача 2. Максимум от функции. Элемент квадрат, которого максимален.

```
if (a * a > max*max)
```

Задача 3. Максимум с условием. Максимальный четный элемент. Нельзя первый элемент в качестве начального элемента.

12.1.1

Если известно ограничение на диапазон значений элементов. Тогда все просто, максимум равен минимуму диапазона+1 до цикла.

12.1.2

Ограничения нет на диапазон. Сначала нужно найти первый элемент удовлетворяющий условию и его взять в качестве начального значения. Потом остальные сравниваем как обычно.

Задача 4. *Найти максимальный четный элемент.*

```
int max = 1;
for (int i = 0; i < n; i++){
    int a = in.nextInt();
    if (a % 2 == 0){
        if(max == 1 || a > max){
            max = a;
        }
    }
}
if (max == 1)
    out.println("NO");
else
    out.println(max);
```

12.2 Цикл с условием

12.2.1 Цикл с предусловием.

Сначала проверяет условие, потом делает тело цикла или выходит из цикла, потом проверяет условие.

```
while (cond) {
    body;
}
```

12.2.2 Цикл с предусловием

Сначала делает тело цикла, потом проверяет условие.

```
do {
    body;
}while(cond);
```

12.3 Примеры

Задача 5. *Дана последовательность целых чисел, которая заканчивается числом 100. Найти сумму чисел, больше 20.*

12.3.1 Последний элемент не нужно учитывать.

```
int a = in.nextInt();
int sum = 0;
while(a != 100){
    if (a > 20){
        sum += a;
    }
    a = in.nextInt();
}
out.println(sum);
```

```
int sum = 0;
int a = in.nextInt();
if (a != 100){
    do {
        a = in.nextInt();
        if (a > 20) {
            sum += a;
        }
    }while(a != 100);
}
out.println(sum);
```

12.3.2 Последний элемент нужно учитывать

```
int sum = 0;
int a;
do {
    a = in.nextInt();
    if (a > 20){
        sum += a;
    }
}while(a != 100);
```

```
int a = 1;
int sum = 0;
while (a != 100){
    a = in.nextInt();
    if (a > 20){
        sum +=a;
    }
}
```

12.3.3 Еще пример

Задача 6. Дано натуральное число x , найти количество единиц в троичном представлении числа.

```
int x = in.nextInt();
int k = 0;
while (x != 0){
    if (x % 3 == 1){
        k++;
    }
    x /= 3;
}
```

13.10.22

13 Массивы.

Массивы - совокупность однотипных данных, имеющих общее имя, при этом каждый элемент имеет уникальный номер, который называется индексом. Данные в массиве надо обрабатывать с помощью цикла.

13.1 Описание и создание массива

13.1.1 Описание

```
DataType [] nameArray;
```

```
int [] a;
```

13.1.2 Создание

```
nameArray = new DataType[length];
```

```
a = new int[10]
```

13.1.3 Прошлые действия в одном

```
int []a = new int[10];
```

13.2 Заполнение массива.

Обнулим массив.

```
for(int i = 0; i < a.length;i++){
    a[i] = 0;
}
```

Заполним элементами последовательности.(2,4,6,8)

```
int[0] = 2;
for(int i = 1; i < a.length; i++){
    a[i] = a[i - 1] + 2;
}
```

Ввод элементов массива с клавиатуры.

```
for (int i = 0; i < a.length; i++){
    a[i] = in.nextInt();
}
```

Если в квадратных скобках пишем что-то кроме `i`, то проверяем не вышли ли из границ массива.

13.3 Вывод массива на экран.

```
for (int i = 0; i < a.length ; i++){
    out.print(a[i] + " ");
}
out.println();
```

13.4 Подсчет элементов

Сумма отрицательных элементов

```
int sum = 0;
for (int i = 0; i < a.length; i++){
    if (a[i] < 0){
        sum += a[i];
    }
}
out.println(sum);
```

14 Перестановки

Пусть есть две ячейки памяти, надо поменять их местами.

```
int x = 10;
int y = 5;
int z = x;
x = y;
y = z;
```

15 Поиск минимума и максимума.

Можно не хранить значение переменной `max` без условия.

```
int imax = 0;
for (int i = 1 ; i < a.length; i++){
    if (a[i] > a[imax])
```

```
        imax = i;
    }
```

С условием. Есть тонкости

1. Нельзя 0 элемент считать значением максимума
2. Если известно ограничение на диапазон, в качестве начального значения берем число, выходящее за диапазон.
3. Если неизвестно, то надо найти первый подходящий элемент.

16 Заполнение массива случайными числами

1. Подключаем библиотеку Random

```
import java.util.Random
```

2. Создать объект класса Random

```
Random rnd = new Random(0);
```

3. Используем этот объект

```
rnd.nextInt(k);
```

Выдает случайное целое число от 0 до k-1.

17 20.10.20

17.1 Однопроходные алгоритмы.

Перед тем, как сохранить входные (и текущие) данные в массив, нужно ответить на вопрос "нужно ли проходить по данным больше двух раз". Если нужно, то храним, иначе не храним.

17.2 Обработка строк

Строка(String) - некоторое количество символов, стоящее в ряд.

```
String s = "jopa";
```

Строки неизменяемый тип данных.

```
s = s + " piska";
```

Создается новая строка.

```
s.length();
```

Длина строки.

```
String h = "";
```

Пустая строка.

```
s.substring(begin)
```

Подстрока начиная с позиции begin. Индексы элементов строки начинаются с нуля.

```
s.substring(begin, end);
```

Часть строки с begin до end (не включая).

```
s.indexOf(что);
```

Номер позиции первого вхождения что в строку s. Результат равен -1, если такого нет.

```
s.charAt(n)
```

Символ строки с указанным номером.

Строки нельзя сравнивать операциями сравнения.

```
s.equals(s1);
```

Равенство строк.

```
s.compareTo(s1);
```

Эта штука сравнивает в лексикографическом порядке.

```
in.nextLine();
```

Ввод строки с клавиатуры.

17.2.1 Пример

Дана строка и число. Хотим считать с клавиатуры.

```
int x = in.nextInt();  
String s = in.nextLine();
```

Не будет работать.

```
int x = in.nextInt();  
in.nextLine();  
String s = in.nextLine();
```

Будет работать.

18 27.10.22

18.1 Подпрограммы(статические методы)

18.2 Подпрограммы без параметров

```
public static returnType name() {  
    body;  
    return result;  
}
```

void - подпрограмма ничего не возвращает. Хороший стиль программирования - ровно один return в последней строке.

18.2.1 Вызов подпрограммы.

```
name();
```

18.2.2 Примеры

Задача 7. Вводим двухзначное число, возвращаем Σ цифр.

```
public static int sumDigits(){
    int x = in.nextInt();
    return x/10 + x%10;
}
```

Задача 8. Вводим целое число, проверяем на четность.

```
public static isOdd() {
    int x = in.nextInt();
    return x%2 != 0;
}
```

Задача 9. Ввести имя и вывести приветствие.

```
public static void sayHello() {
    String name = in.nextLine();
    out.println("Hello, "+name+"!!!!!!!!!!");
}
```

18.3 Подпрограммы с параметрами.

```
public static typeResult name(type name, type name){
    body;
}
```

18.3.1 Пример

Задача 10. Найти максимум двух целых чисел.

```
static int max(int a, int b) {
    int result;
    if (a > b){
        max = a;
    }
    else{
        max = b;
    }
    return result;
}
```


Параметры передаются по значению.

```
f(x)
```

x не меняется точно.

Задача 11. Подпрограмма заполнения массива четными числами от двух.

```
static void fillArray(int [] a) {  
    a[0] = 2;  
    for (int i = 1; i < a.length(); i++){  
        a[i] = a[i - 1] + 2;  
    }  
}
```

19 03.11.2022

19.1 Рекурсия.

Рекурсия - это обращение к самому себе.

Задача 12. Факториал.

```
static int fac(int n) {  
    if (n == 1) {  
        return 1;  
    }  
    return n * fac(n - 1);  
}
```

19.2 Структура рекуррентной программы

1. Проверка условия продолжения рекурсии.
2. Если условие выполнилось
 - а) Действие
 - б) Вызов самой себя.

Задача 13. С клавиатуры вводится последовательность целых чисел, которая кончается нулем. Посчитать количество четных чисел.

```
static int count() {  
    int x = in.nextInt();  
    if (x == 0) {  
        return 0;  
    }  
  
    else {  
        if (x % 2 == 0) {
```

```
        return 1 + count();
    }
    return count();
}
}
```

Задача 14. Вывести на экран вертикально данно числа десятичные числа.

```
static void digit(int n) {
    if (n > 0) {
        digit(n / 10);
        out.println(n % 10);
    }
}
```

20 10.11.2022

20.1 Объектно-ориентированное программирование.

Три кита (парадигмы) ОПП:

- Абстрация
- Инкапсуляция
- Наследование
- Полиморфизм