Переменные
Примитивные типы
Операторы

правила объявления переменных,
работа с примитивными типами,
приведение примитивных типов и тд

Данные в Java

Каждая программа в конечном итоге сводится к манипулированию данными.

Типы данных, используемых в Java:

Примитивные типы данных:

1) byte (целые числа, 1 байт)
2) short (целые числа, 2 байта)
3) int (целые числа, 4 байта)
4) long (целые числа, 8 байтов)
5) float (вещественные числа, 4 байта)
6) double (вещественные числа, 8 байтов)
7) char (символ Unicode, 2 байта)
8) boolean (значение истина/ложь, 1 бит / либо int)

Ссылочные типы данных:

1) String
2) массивы
3) классы
4) интерфейсы

Переменные

Для того, чтобы осуществлять манипуляции с данными, используя возможности языка, программа должна сначала где-то сохранить эти данные.

Данные хранятся в переменных.

Переменные - поименованная выделенная область памяти.

Переменные в Java

В языке Java все **переменные должны быть объявлены, перед тем, как они будут использоваться**.

Объявление переменных состоит из указания типа данных, которые будут храниться в переменной и имени этой переменной. По имени переменной и осуществляется доступ к данных, которые в ней хранятся.

типДанных имяПеременной; Например, **int** messageId;

Переменные в Java

Требования в именам переменных (обязательные):

1. имя переменной должно начинаться с буквы (технически может начинаться со $ или _)
2. имя переменной должно состоять из букв (Unicode), цифр и символа _
3. пробелы при именовании переменных не допускаются
4. имя переменной не должно быть ключевым или зарезервированным словом
5. имя переменной чувствительно к регистру

Требования в именам переменных (по соглашению Java Code Conventions)**:**

1. запрещено начинать имя переменной со _ и $.
2. символ доллара «$», по соглашению, никогда не используется
3. имя переменной должно начинаться именно с маленькой буквы
4. при выборе имен переменных, следует использовать полные слова.
5. если имя переменной состоит из более чем одного слова, то отделяйте каждое последующее слово в имени переменной заглавной буквой

Переменные в Java

После того, как переменная была объявлена, с ней можно работать, например, **присвоить значение**:

```
типДанных имяПеременной;
имяПеременной = значение;
```

Можно объявить сразу несколько переменных одного типа:

```
типДанных имяПеременной1, имяПеременной2, имяПеременной3;
```

Можно присвоить значения сразу нескольким переменным:

```
типДанных имяПеременной;
 имяПеременной1 = имяПеременной2 = имяПеременной3;
```

Можно присвоить значение в момент объявления переменной;

```
типДанных имяПеременной = значение;
```

Целочисленные типы данных в Java

Представлены типами byte, short, int, long.
Могут быть положительными и отрицательными.

Тип byte (8 бит) - наименьший по размеру целочисленный тип.
Диапазон допустимых значений от -128 до 127.

Переменные типа byte **часто используются**

- при работе с потоком данных из сети или файла,
- при работе с необработанными двоичными данными
- массивах для экономии памяти.

Целочисленные типы данных в Java

Тип short (16 бит) используется крайне редко
Диапазон допустимых значений от -32768 до 32767.

В арифметических выражениях с переменными типа short вычисления выполняются как с типом int, т.е. с помощью 32-битовой арифметики, а полученный результат будет 32-битовым.

Например,

```
short a = 2;
short b = 3;
short c = a + b;
```

в этой строке будет **ошибка**, в арифметических выражениях с переменными типа short вычисления выполняются как с типом int и **результатом будут данные типа int**

Целочисленные типы данных в Java

Тип int (32 бита) самый распространенный целочисленный тип данных **Диапазон допустимых значений от -2147483648 до 2147483647.**

В Java 7 можно использовать знак подчеркивания для удобства.

Например,
```
int c = 1_000_000;
```

При делении целочисленных чисел остаток отбрасывается (8 / 3 = 2).

Деление на ноль приведет к ошибке.

Целочисленные типы данных в Java

Тип long (64 бита) используется для хранения очень больших значений.
Диапазон допустимых значений
от -9223372036854775808 до 9223372036854775807.

Можно использовать символы l или L для обозначения числа типа long (L использовать более предпочтительно, ее сложнее спутать с 1).

Например,
long x = 6000000000; воспринимает, как int, сообщая, что это число слишком большое
long x = 6000000000L; воспринимает, как long

Типы с плавающей точкой в Java

Применяются при вычислении выражений, в которых требуется точность до десятичного знака.
float и double представляют числа одинарной и двойной точности.

При делении данных типов на 0 получаем бесконечность (Infinity)

Тип float(32 бита) определяет значение одинарной точности.
Диапазон допустимых значений от -3.4E+38 до 3.4E+38.

Используются, когда требуется дробная часть без особой точности.

Рекомендуется добавлять символ F или f для обозначения этого типа, иначе число будет считаться типом double.
Например,

```
float x = 11.6F;
```

Для точных вычислений дробной часть используйте, например, класс BigDecimal

Типы с плавающей точкой в Java

Тип double(64 бита) определяет значение двойной точности.
Диапазон допустимых значений от -1.7E+308 до 1.7E+308.

Современные процессоры оптимизированы под вычисления значений двойной точности, поэтому они предпочтительнее, чем тип float.

Для точных вычислений дробной часть используйте, например, класс BigDecimal

Символы в Java

Тип char(16 бит) используется для хранения символов.

Диапазон допустимых значений от '\u0000' до '\uffff' (от 0 до 65535).

В Java для char используется кодировка Unicode.

```
char ch1, ch2, ch3, ch4;
ch1 = 74; // код переменной
ch2 = 'a'; // сам символ
ch3 = 118; // код переменной
ch4 = 'a'; // сам символ
System.out.println(ch1 + ch2 + ch3 + ch4); // Java
```

Переменной можно присвоить код символа или сам символ в одинарных кавычках..

Символ 'a' не строка "a"

Стандартные символы ASCII можно выводить сразу.
Если нужно вывести специальный символ из Unicode, то можно воспользоваться шестнадцатеричным представлением кода в escape-последовательности - обратный слеш и четыре цифры после u.

Например, **char** ch = **'\u0054'**;

Булевые значения в Java

Тип boolean предназначен для хранения логических значений и может принимать только одно из двух возможных значений:

- true
- false.

Данный тип всегда возвращается при использовании операторов сравнения

Также он используется в управляющих конструкциях, например if.

```
boolean isActive = true;

boolean started = false;
```

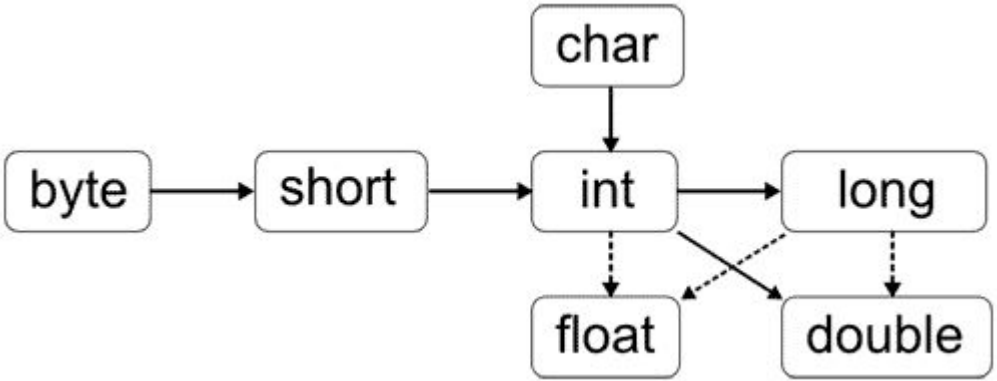
Приведение примитивных типов в Java

В Java существует 2 типа преобразований:

1. Автоматическое преобразование;
2. Приведение типов.

Автоматическое преобразование типов осуществляется, если:
оба типа совместимы и длина целевого типа больше длины исходного типа.
В этом случае происходит преобразование с расширением.

Например,
```
byte a = 15;
int b = a;    // a автоматически преобразовано к int
```



Сплошные линии *обозначают преобразования, выполняемые* ***без потери данных****.*
Штриховые линии *говорят о том, что при преобразовании* ***может произойти потеря точности.***

Приведение примитивных типов в Java

Если длина целевого типа меньше длины исходного типа, необходимо использовать **явное приведение типов**.

При явном приведении типов перед целевым значением необходимо в круглых скобках указать целевой тип данных: **(целевой тип) целевое значение**.

Например, при преобразовании типа int к типу long:

```
int i = 2015;   => long l = (long) (i);
```

Правила приведения типов:

- При приведении float или double к целочисленным типам, дробная часть не округляется, а просто отбрасывается.
- Тип boolean не приводится ни к одному из типов.
- Тип char приводится к числовым типам, как код символа в системе UNICODE.
- Если число больше своего контейнера, результат будет непредсказуемым.

Математические операторы в Java

Бинарные арифметические операции (производятся над двумя операндами)

+	сложение
-	вычитание
*	умножение
/	деление
%	взятие остатка от деления

Математические операторы в Java

Бинарные арифметические операции (некоторые правила):

1. Если в операции участвуют два целых числа, то результат деления будет округляться до целого числа. Чтобы результат представлял число с плавающей точкой, один из операндов также должен быть числом с плавающей точкой, либо для результата необходимо использовать приведение типов.
2. Если один операнд имеет тип long, то тип всего выражения повышается до long.
3. Если один операнд имеет тип float, то тип всего выражения повышается до float.
4. Если один операнд имеет тип double, то тип всего выражения повышается до double.

Математические операторы в Java

Унарные арифметические операции (производятся над одним операндом)

Инкремент и **декремент**: Увеличивают / уменьшают на 1

i++	инкремент (постфиксная форма) увеличивает, и возвращает старое значение
++i	инкремент (префиксная форма) сначала увеличивает, а потом возвращает значение
i--	декремент (постфиксная форма) уменьшает, и возвращает старое значение
--i	декремент (префиксная форма) сначала уменьшает, а потом возвращает значение

Математические операторы в Java

Оператор сложения (+) используется для:

1. сложения чисел;
   Например, 2 + 6; // 8
2. конкатенации (склеивание) строк;
   Если хотя бы один аргумент является строкой, то второй будет также преобразован к строке, после чего конкатенация слияние строк
   Например, '2' + 6; // '26'
3. приведения значения к типу int,
   если это переменная типа byte, short или char

Операторы сравнения в Java

В операциях сравнения сравниваются два операнда, и возвращается значение типа boolean:

true, если выражение верно, false, если выражение неверно.

>	больше
<	меньше
==	равно
>=	больше или равно
<=	меньше или равно
!=	не равно

Логические операторы в Java

Представляют условие и возвращают true или false.

Операторы и случаи, в которых они возвращают true

&&	a && b	a и b истинны, b оценивается условно (если a ложно, b не вычисляется)
||	a || b	a или b истинно, b оценивается условно (если a истинно, b не вычисляется)
!	!a	a ложно
&	a & b	a и b истинны, b оценивается в любом случае
|	a | b	a или b истинно, b оценивается в любом случае
^	a ^ b	либо a, либо b (но не одновременно) равны true

Операторы присваивания в Java

=	a = 12	Переменной a присвоили значение 12
+=	a += b	Краткая форма записи a = a + b (сложение с присваиванием)
-=	a -= b	Краткая форма записи a = a – b (вычитание с присваиванием)
*=	a *= b	Краткая форма записи a = a * b (умножение с присваиванием)
/=	a /= b	Краткая форма записи a = a / b (деление с присваиванием)
%=	a %= b	Краткая форма записи a = a % b (деление по модулю с присваиванием)

Тернарный оператор в Java

состоит из трех операндов и используется для оценки выражений типа boolean. **Цель тернарного оператора** заключается в том, чтобы решить, какое значение должно быть присвоено переменной.

Оператор **записывается в виде**:
переменная x = (условие) ? выражение1 : выражение2;

Если **условие** равно true, то вычисляется **выражение1** и его результат становится результатом выполнения всего оператора.

Если же **условие** равно false, то вычисляется **выражение2**, и его значение становится результатом работы оператора.

Оба операнда выражение1 и выражение2 должны возвращать значение одинакового (или совместимого) типа.