JAVA PLATFORM

Компиляция – перевод данных в байт код (000011100)

Интерпретация – Компилирует каждую строчку сразу и передает на процессор

SE (Standard Edition) — платформа, предназначенная для разработки (программирования) приложений, работающих в серверных и десктоп окружениях.
ME (Micro Edition)* — платформа, предназначенная для разработки (программирования) приложений, работающих во встраиваемых и мобильных окружениях.
EE (Enterprise Edition)** — набор спецификаций, предназначенных для Enterprise*** приложений.

Примечание*: это не относится к Android.
Примечание**: в данный момент называется Jakarta EE.
Примечание***: корпоративное приложение, используемое крупными компаниями для автоматизации сложных бизнес-процессов.

JVM (Java Virtual Machine) — программа, которая исполняет байт-код.
JRE (Java Runtime Environment) — JVM + стандартная библиотека (готовые, уже написанные для нас компоненты) + ряд инструментов.
JDK (Java Development Kit) — JRE + инструменты разработки (компилятор и другие).

Переменные — это просто удобные имена для хранения наших данных.
Т.е. мы сами придумали их для того, чтобы нам через неделю, месяц и даже пару лет **было понятно, какие данные и для чего там хранятся**.

Правильно подобранные переменные позволяют сделать программу поддерживаемой — при необходимости изменения будет понятно, что и по какой логике изменяется.

Свойства переменных:

Объявление — это сообщение компилятору о том, что мы хотим определить в нашем приложении некоторое имя.
Для того, чтобы объявить переменную, нужно указать тип и имя. Объявление завершается символом ;.

1. **Имя** — определяет, как обращаться к данным, хранящимся по этому имени. Воспринимайте это как название определённой ячейки в таблице.
2. **Тип данных** — то какие данные можно по этому имени. Нужно для того, чтобы случайно в количество (которое, например, в штуках) не положить неправильных данных: строку много. Компилятор сам будет следить за этим.
3. **Сами данные** — то, какие данные сейчас хранятся. Это значит, что данные можно изменять (так же, как данные в ячейке таблицы).

Java — язык со строгой типизацией. Это значит, что создавая переменную, вы должны обязательно указать тип, чтобы в дальнейшем компилятор смог отследить все попытки «положить» в эту переменную «неправильные данные».

Типы в Java делятся на примитивные и ссылочные (будем проходить позже).

- `byte` — 1 байт, от -128 до 127 (-2^7 до 2^7);
- `char` — 2 байта, от 0 до 65 535 ($2^{16} - 1$);
- `short` — 2 байта, от -32 768 до 32 767 (-2^{15} до 2^{15});
- `int` — 4 байта, от -2 147 483 648 до 2 147 483 647 (-2^{32} до 2^{32});
- `long` — 8 байт, от -9 223 372 036 854 775 808 до 9 223 372 036 854 775 807 (-2^{64} до 2^{64}).

Вещественные типы предназначены для хранения вещественных чисел (чисел с дробной частью, например, 3.14).

К вещественным относят два типа:

- `float` — 4 байта, от $1.40239846 * 10^{-45}$ до $3.40282347 * 10^{38}$;
- `double` — 8 байт, от $4.9406564584124654 * 10^{-324}$ до $1.7976931348623157 * 10^{308}$.

Проще запомнить, что `double` может хранить в два раза больше значащих цифр, чем `float`.

Для того, чтобы запустить программу в режиме отладки, необходимо сделать следующее:

1. Создать конфигурацию запуска (`Ctrl + Shift + F10`).
2. Поставить точку остановки (`Ctrl + F8`) — точку, в которой отладчик остановит выполнение программы, чтобы продолжить по шагам.

Вспомним, что к примитивным типам относятся:

- boolean;
- byte, char, short, int, long;
- float, double.

Ключевые клавиатурные сокращения*:

1. Alt + Insert в панельке Project — генерация файлов.
2. main + Tab в редакторе кода — генерация public static void main....
3. Ctrl + Shift + F10 — генерация конфигурации запуска и запуск приложения.
4. Ctrl + Alt + L — выравнивание кода.
5. Ctrl + F8 — установка/снятие точки остановки.
6. Shift + F9 — запуск под отладчиком.
7. F8 — исполнение следующей “строки” в режиме отладки.

На Mac OS посмотреть клавиатурные сокращения можно через пункт меню Help -> Keymap Reference.

ЛИТЕРАЛЫ

Литерал — это непосредственное значение, записанное в коде.

Например, в выражении: `float amount = 1000.50`, `1000.50` — это литерал.

В Java к численным литералам применяются два ключевых правила:

- если литерал представляет из себя целое число, то он имеет тип `int`
- если литерал представляет из себя вещественное число (число с точкой), то он имеет тип `double`

Другой вопрос, если в выражении встречаются разные типы данных, то используются чёткие правила для выведения типа всего выражения (т.к. результатом выражения может быть только одно значение):

1. Если один из операндов `double`, то всё выражение `double`.
2. Если один из операндов `float`, то всё выражение `float`.
3. Если один из операндов `long`, то всё выражение `long`.
4. Всё остальное — `int`.

Таким образом, например, `long + float` будет `float`.

ПРИВЕДЕНИЕ ТИПОВ

Q: хорошо, а что если нужно сделать из `float` `int`?

A: для этого есть специальный оператор `cast`:

```
int result = (int)99.99F;
```

При этом произойдёт принудительное преобразование из `float` в `int`.

Естественно, вместо `int` можно подставить любой тип.

ОБЛАСТЬ ВИДИМОСТИ

{} определяют блок кода, ограничивающий область видимости переменных.

Область видимости переменной — это область, в которой переменная доступна по имени.

Это значит, что переменная не доступна за пределами области видимости, в которой она объявлена.

ОПЕРАТОРЫ СРАВНЕНИЯ

Теперь, когда мы знаем как использовать конструкцию if, осталось только добавить лимит. В этом нам помогут операторы сравнения:

- == — проверка на равенство (не используйте с вещественными числами);
- != — проверка на неравенство;
- < — меньше;
- <= — меньше или равно;
- > — больше;
- >= — больше или равно.

Важно: операторы сравнения всегда возвращают boolean.
Вам, как тестировщикам стоит помнить, что именно эти операторы определяют границы, соответственно, создают ошибки граничных значений.

ТЕРНАРНЫЙ ОПЕРАТОР

Иногда для сокращения кода используют тернарный оператор.
Выглядит он следующим образом:

```
expression ? if-true-value : if-false-value
```