ПЕРЕМЕННЫЕ И ТИПЫ ДАННЫХ

AIT
Hi-tech team

НАШИ ПРАВИЛА

Включенная камера

Вопросы по поднятой руке

Не перебиваем друг друга

Все вопросы, не связанные с тематикой курса (орг-вопросы и т. д.), должны быть направлены куратору

Подготовьте свое рабочее окружение для возможной демонстрации экрана (закройте лишние соцсети и прочие приложения)

ЦЕЛЬ

Изучить способы работы с переменными и особенности различных типов данных

ПЛАН ЗАНЯТИЯ

- Представление информации и переменные
- int
- double
- char
- Преобразование типов
- Операции

ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ

Компьютер хранит информацию в двоичном (бинарном) коде

Для того, чтобы правильно интерпретировать эту информацию, необходимо знать ее тип (например, целое число, символ и т.д.)

Каждое двоичное значение хранится в одном бите и представляет собой либо 1 (истина, true) либо 0 (ложь, false)

Бинарный код основан на двоичной системе счисления, тогда как мы используем десятичную систему.

Группа из 8 бит образует 1 байт. Максимально число, которое можно выразить 1 байтом - 255 (почему?)

7 6 5 4 3 2 1 0

$$10001001_2 = 1\cdot2^7+0\cdot2^6+0\cdot2^5+0\cdot2^4+1\cdot2^3+0\cdot2^2+0\cdot2^1+1\cdot2^0 = 137_{10}$$

ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ

Для того, чтобы представить десятичное число в двоичной системе счисления, можно воспользоваться вычислением остатков от деления на 2 и “прочесть” полученные значения снизу вверх

137 / 2 = 68 остаток: 1
68 / 2 = 34 остаток: 0
34 / 2 = 17 остаток: 0
17 / 2 = 8 остаток: 1
8 / 2 = 4 остаток: 0
4 / 2 = 2 остаток: 0
2 / 2 = 1 остаток: 0
1 / 2 = 0 остаток: 1

итог: 10001001_2

ДВОИЧНАЯ АРИФМЕТИКА

Операции сложения и вычитания для двоичной системы счисления аналогичны этим операциям в десятичной системе счисления:

$$0_2 + 0_2 = 0_2 = 0_{10}$$

$$0_2 + 1_2 = 1_2 = 1_{10}$$

$$1_2 + 1_2 = 10_2 = 2_{10}$$

$$1_2 + 1_2 + 1_2 = 11_2 = 3_{10}$$

Сложение двоичных чисел “столбиком”

$$\begin{array}{rr} & {\scriptstyle 1\,1\qquad 1\,1\,1} \\ & 11100111 \\ + & 11100111 \\ \hline & 111001110 \end{array}$$

ПРЕДСТАВЛЕНИЕ ИНФОРМАЦИИ

Для хранения отрицательных чисел компьютер использует дополнительный код

Представление -5_{10}

$5_{10} = 00000101_2$ (прямой код)

обратный код: 11111010_2

дополнительный код: 11111011_2

Для того, чтобы понять, что байт хранит отрицательное число, старший разряд используется как знак (1 - отрицательное число в дополнительном коде, 0 - положительное в прямом коде)

11111011_2

Следовательно, с учетом знака, максимальное число для 1-го байта 127, минимальное -128 (почему?)

СТАВИМ ПЛЮС, ЕСЛИ ПОНЯТНО

- Компьютер хранит информацию в двоичном коде
- Байт - это набор 8 бит
- Механизмы преобразования чисел из десятичной системы счисления в двоичную и обратно
- Основы двоичной арифметики
- Способ хранения отрицательных чисел

ПЕРЕМЕННЫЕ

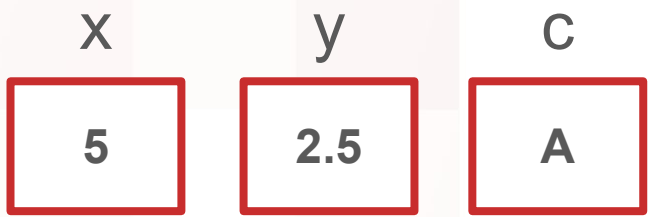
Переменная представляет собой участок памяти, которому присвоено определенное **имя**

Чтобы использовать переменную, ее необходимо **объявить**, указав имя и тип

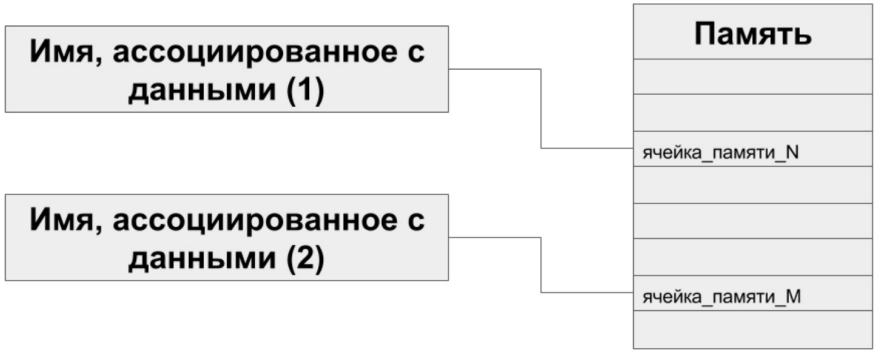
```
int x = 5;
double y = 2.5;
char c = ‘A’
```

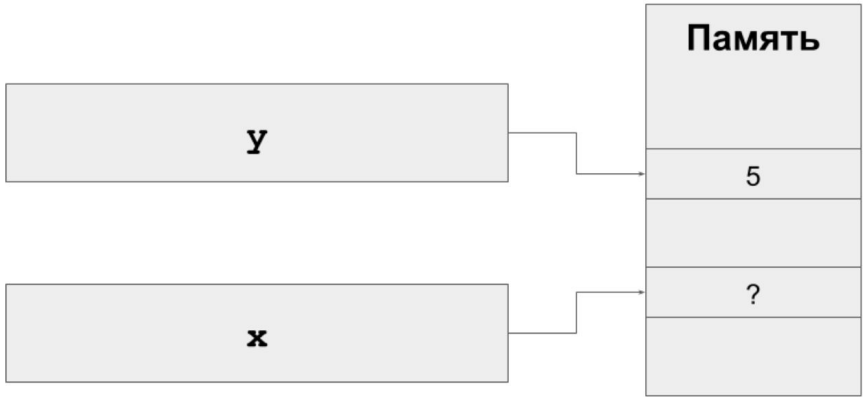
Переменная содержит данные определенного **типа** - целое число, вещественное число, символ и т.д.

Переменной можно **присвоить** какое-либо значение, соответствующее ее типу



ПЕРЕМЕННЫЕ


Имя, ассоциированное с данными (1)
Имя, ассоциированное с данными (2)
Память
ячейка_памяти_N
ячейка_памяти_M


y
x
Память
5
?

int

int - целочисленный тип данных, занимает 4 байта

Минимальное значение -2147483648, максимальное 2147483647

С переменными целого типа возможно выполнение следующих операций - +, -, *, /, %

Результат деления переменных целого типа всегда будет целочисленным

```
int a = 13;
int b = 3;
int x = a / b; // 4
int y = a % b; // 1
```

double

double - вещественный тип данных, занимает 8 байт

Минимальное положительное - $4.9 * 10^{-324}$, максимальное положительное значение - $1.7976931348623157 * 10^{308}$

```
double a = 7;
double b = 2;
double c = a / c; // 3.5
```

```
int a = 7;
int b = 2;
double c = a / c; // 3.0
```

char

char - символьный тип, занимает в памяти 2 байта

```
char upper = ‘A’;
char lower = ‘a’;
char cyrillic = ‘Ы’;
```

Представляет собой числовое значение в диапазоне от 0 до 65 535, которое ассоциировано с символом из таблицы Unicode

```
char d = ‘8’;
int number = d - ‘0’; // d = 8
```

ПРИМИТИВНЫЕ ТИПЫ ДАННЫХ

Целочисленные

byte
int
short
long

Символьные

char

Вещественные

float
double

Логические

boolean

ПРИМИТИВНЫЕ ТИПЫ ДАННЫХ

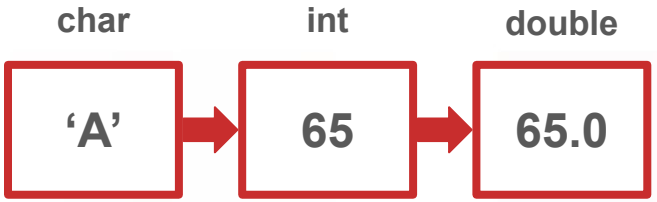
Тип данных	Размер (бит)	Значения
byte	8	от -128 до 127
short	16	от -32768 до 32767
char	16	от 0 до 65535
int	32	от -2147483648 до 2147483647
long	64	от -9223372036854775808 до 9223372036854775807
float	32	от -1.4e-45f до 3.4e+38f
double	64	от -4.9e-324 до 1.7e+308
boolean	1 или 32	true или false

СТАВИМ ПЛЮС, ЕСЛИ ПОНЯТНО

- Примитивные типы данных и их назначения
- Особенности работы с примитивными типами
- Литеральные значения и переполнение
- Основы двоичной арифметики
- Способ хранения отрицательных чисел

ПРЕОБРАЗОВАНИЯ ТИПОВ

Неявное преобразование выполняется автоматически из переменных типов с меньшим диапазоном значений в переменные типов с бОльшим диапазоном значений



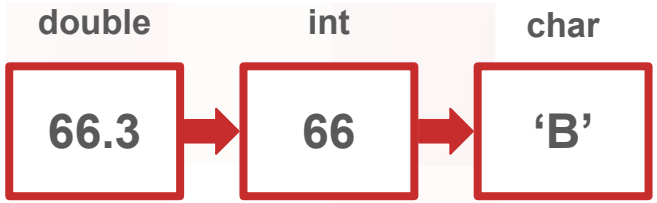
Такое преобразование выполняется автоматически, поскольку потеря данных невозможна.

ПРЕОБРАЗОВАНИЯ ТИПОВ

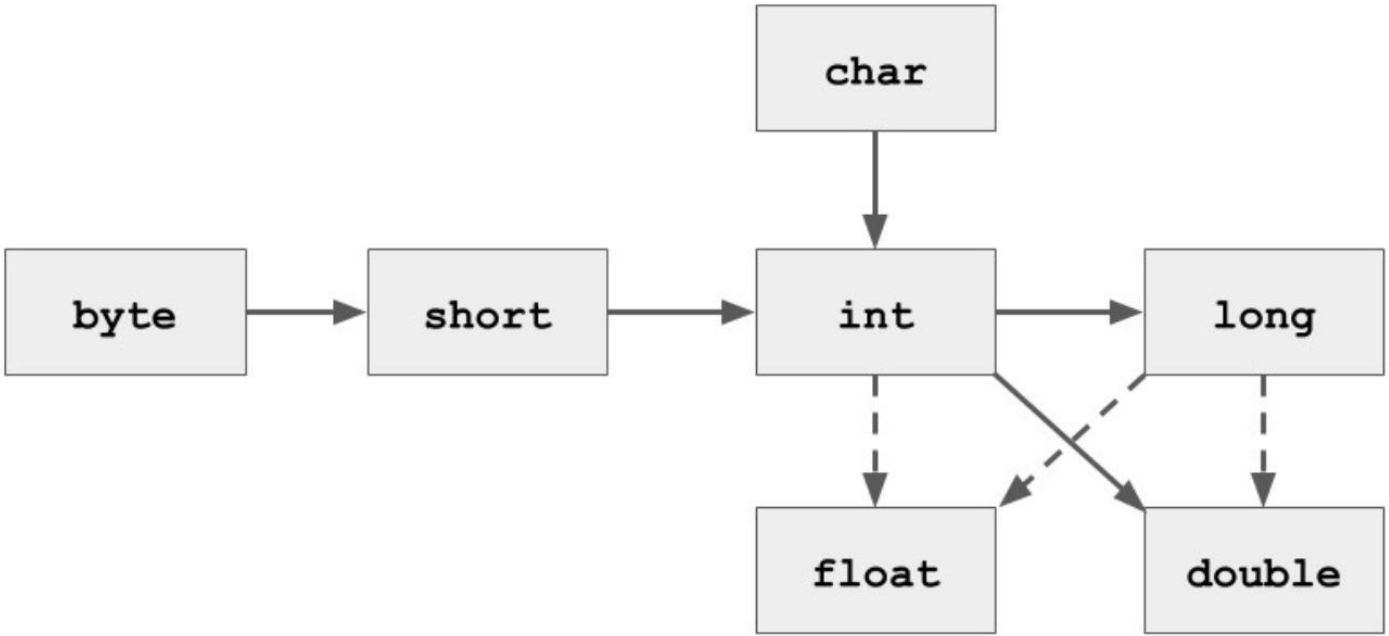
Явное преобразование выполняется с явным указанием целевого типа и применяется в обратной ситуации

```
double x = 66.3;
int y = (int)x;
char z = (char)y;
```

Такое преобразование выполняется автоматически, поскольку потеря данных невозможна.



ПРЕОБРАЗОВАНИЯ ТИПОВ



СТАВИМ ПЛЮС, ЕСЛИ ПОНЯТНО

Неявное преобразование

Возможные потери данных при явном преобразовании

ТЕРМИНАЛОГИЯ

Символы — набор знаков, допустимых для словообразования (алфавит) в данном языке. Применимо к языкам программирования — это множество знаков, которые могут быть использованы при написании исходного кода программы.

Лексемы — слова, образованные из алфавита заданного языка. Являются минимальной смысловой единицей в тексте. Применимо к языкам программирования — это последовательности символов, являющиеся именами переменных, ключевыми словами языка либо специальными или зарезервированными последовательностями символов.

Выражения — множество слов (или лексем), которые несут в себе некоторую законченную мысль. В естественных языках применим термин — предложения. В языках программирования примером предложения (выражения) может быть следующее:

```
x = v * t + x0;
```

Операторы — в общем случае, служебные элементы логического языка. В естественных языках к ним можно отнести артикли. Применимо к языкам программирования — это служебные последовательности символов, применяемые к одному или нескольким операндам в составе выражений.

ОПЕРАЦИИ

Унарные	Бинарные
-	- +
--	
++	/ *
~	
!	| ||
[]	
	& &&
	== !=

```
int x = -y;          // унарный “минус”

int r = x - y;       // бинарный “минус”
```

ОПЕРАЦИИ

№	Оператор
1	[] . ()
2	! ~ ++ -- + - (приведение) new
3	* / %
4	+ - (бинарные)
5	>> << >>>
6	< <= > >= instanceof
7	== !=
8	&
9	^
10	|
11	&&
12	||
13	?:
14	= += -= *= /= %= |= ^= <<= >>= >>>=

СТАВИМ ПЛЮС, ЕСЛИ ПОНЯТНО

Терминология

Операции

СТРОКИ

```
String name = "Alis";
```

```
String lastName = new String("Parker");
```

```
String a = "Java\u2122";          // Java™
String b = "";                    // пустая строка
```

СТРОКИ

Конкатенация	str.concat("!")
Длина строки	str.length()
Деление строки по критерию	str.split(", ")
Получение символа по индексу	str.charAt(5)
Проверка наличия подстроки	str.contains("мир")
Проверка пустой строки	str.isEmpty()
Замена символов строки	str.replace("Привет", "Здоров")
Усечение пробельных символов	str.trim()
Вычленение подстроки	str.substring(0, 5)
Получение массива символов	str.toCharArray()

ПОИГРАЕМ ;)

- Переменная
- Тип данных
- double
- Явное преобразование
- Литерал

ДОМАШНЕЕ ЗАДАНИЕ

1. попробуйте перевести десятичные числа в двоичные и обратно (в том числе и отрицательные), и осуществите арифметические операции (сложения и вычитания достаточно) с двоичными числами

2. дописать CurrencyConverter, добавив считывание пользовательского ввода, то есть добавьте возможность ввода изначальной валюты (посмотрите как использовать java.util.Scanner - необходим для считывания пользовательского ввода)

AIT
Hi-tech team

Ваша новая IT-профессия – Ваш новый уровень жизни

Программирование с нуля в
немецкой школе AIT TR GmbH

AIT
Hi-tech team