

Java_Class{

class-level data (variables);

...

class-level functions (...) {

...

};

}

Java variables:

- have to have specified type (either class or primitive type)
- have to start with letter
- are not allowed to have special characters (;,.,etc.)
- have to be different from keywords

Java primitive types:

- boolean
- char (Unicode character, strings)
- integers (byte, short, int, long)
- floats (float, double)

Integer numbers:

- decimal numbers (not used by computers)
 $123_{10} = 1 \cdot 10^2 + 2 \cdot 10^1 + 3 \cdot 10^0$
- binary numbers (used by computers)
 $101_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 5$
- maximum number that can be combined from
N “binary digits” = 1...1 (N ones) = $1 \cdot 2^{N+1} - 1$
 - 8 digits (byte): ~ 127
 - 16 digits (short) : 32,000
 - 32 digits (int): $\sim 2 \cdot 10^9$
 - 64 digits (long) $\sim 10^{18}$

Integer arithmetic and negative numbers:

- two numbers are added “bit by bit”. If N+1-st bit appears, it is disregarded
- Negative number: start with positive number, invert all bits, and add 1.

Floating point numbers:

- Consist of “mantissa” and “exponent” (both binary)
 $(-1)^s \times \text{mantissa} \times 2^{\text{exp}}$
- Assume that is mantissa scaled so that the decimal point is after first “1”
 - This first “1” is assumed, disregarded in computer to save space
- Mantissa has a “sign” bit
- Exponent is stored in the form “*unsigned integer*”-”*shift*”,
size of exponent and shift depend on type of float (single, double precision)

Types of floats in Java:

- float: 4 bytes, (1 sign bit, 8 bits exponent, 23 bit mantissa)
 - range: $2^{-149} \sim 10^{-45} \dots 2^{128} \sim 10^{38}$
 - precision: $\sim 2^{-23} \sim 10^{-7}$
- double: 8 bytes, (1 sign bit, 16 bit exponent, 47 bit mantissa)
 - range: $\sim 10^{-324} \dots \sim 10^{308}$
 - precision: $\sim 10^{-16}$
- three types of machine-precision errors: overflow, underflow, truncation

Functions in Java

- defined by **signature**: function name + types and number of parameters
 - note: parameters names are not included in signature
 - note: return type is not included in signature
 - argument coercion: changing the argument type without losing precision
 - function overloading: having several different functions with same name
- every function within class has to have unique signature (so the compiler would be able to select it when needed)
- non-void functions must have return statement

Accessing the data

- **public** functions may be accessible outside of the class
- **private** functions will not be accessible outside of the class
- **static** functions and variables are shared among all objects of the class
 - static functions may access only static data

Hiding the data

- private data may be accessed/modified through public functions (useful for larger projects, separates implementation details from user interface)

Packages

- A number of classes (with their functions) are grouped into packages
- Packages can be accessed via **import** statement. Some packages (System, Math,...) are imported automatically
- To access a function from the package, one must include package name:
Math.sin(Math.PI)...

Flow control: loops

- for(INI, COND, ITER){
 body
}

- execution order: INI, followed by loop through:
COND, body, ITER