J04 Introductory Java 4

Expressions
Operators
Statements
Methods

ANU – School of Computing – Structured Programming 1110 / 1140 / 6710

Expressions

- A construct that evaluates to a **single value**.
- Made up of
  - variables
  - literals
  - operators
  - method invocations
- Compound expressions follow precedence rules
  - Use parentheses (clarity, disambiguation)

Java Operators

- Assignment (=)
- Arithmetic (`+ - * / % += -= *= /= %=`)
- Unary (`+ - ++ -- !`)
- Equality (`== !=`), relational (`> >= < <=`), logical (`&& ||`) and `instanceof`
- Bitwise (`~ & ^ | << >> >>>`)
- Object creation (`new`)

Statements

- A complete unit of execution.
- **Expression** statements (expressions made into statements by terminating with ‘;’):
  - Assignment expressions
  - Use of ++ or --
  - Method call
  - Object creation expressions
- **Declaration** statements
- **Control flow** statements

Block

- Zero or more statements between balanced braces (‘{’ and ‘}’)
- Can be used anywhere a single statement can

Methods

- A function/procedure/subroutine
  - Reusable code to perform a specific task
  - Abstraction: modularity, encapsulation
- In Java, almost all code is in a method (`main`, if not another).
- Methods may take arguments (parameters).
- Methods may return a value.

Method Declaration

A method declaration consists of the following, in order:

- any **modifiers** (`public`, `private`, etc)
- **return type**
- **method name**
- **parameters**, in parentheses
- (any **exceptions** the method may throw)
- the method **body** (code), a block

```
public byte[] getBytes(String charsetName) {
   …
}
```

Parameters (method arguments)

Parameters are the mechanism for passing information from one method to another method (or constructor).

When a method is called, it must be given a list of argument expressions that match the number and types of the method’s parameters.

```
byte[] bytes = myString.getBytes(“UTF-8”);
```

(The semantics of parameter passing are not so simple, and we will come back to them in a later lecture.)

Returning a Value from a Method

The `return` statement exits the current method.

Methods `return` to caller when:

- all statements in method executed, or
- a `return` statement is reached, or
- the method throws an exception (more in a later lecture)

Methods declared `void` do not return a value.

All other methods must return a value of the declared type (or a subclass of the declared type, described later).

Class and Instance methods

A method declared with the `static` modifier is a **class method** (otherwise it is an **instance method**).

- Class methods
  - Can be called without reference to an object
  - But may use **class fields** only.
- Instance methods
  - Must be called on an object (`anObject.aMethod(...)`)
  - May use both class and **instance fields.**