University of Bamberg

Java-Tutorial for ISSS-Students

Chapter 3: Control flow

Chapter 3: Control flow

1. The *if-then(-else)* Statement
2. The *switch* Statement
3. The *(do-)while* Statement
4. The *for* Statement

The *if-then* Statement

- *if (condition) {code}*
  - At first it is tested whether the condition (which has to be a *boolean expression*) is met
  - If the test evaluates to *true*, the code in parentheses is executed
  - If the test evaluates to *false*, the control flow jumps to the end of the *if-then*-statement

The *if-then-else* Statement

- *if (condition) {code 1} else {code 2}*
  - At first it is tested whether the condition (which has to be a *boolean expression*) is met
  - If the test evaluates to *true*, only *code 1* is executed
  - If the test evaluates to *false*, only *code 2* is executed
- *if (condition 1) {code 1} else if (condition 2) {code 2} … else if (condition n) {code n} else {code x}*
  - The statements can be combined into complex constructs
  - As soon as the first condition is met, the corresponding code is executed and the control flow jumps to the end of the whole constuct

Boolean operators in *if-then-else* Statements

- *if (condition a || condition b) {code 1} else {code 2}*
  - One of the conditions has to be met to execute code 1
- *if (condition a && condition b) {code 1} else {code 2}*
  - Both conditions have to be met to execute code 1

The *switch* Statement

- *switch(expression) {*

  case constant1: insruction1; break;

  case constant2: instruction2; break;

  …

  case constantX: instructionX; break;

  default: instruction;

  }
- The given expression (allowed expressions are: *Strings*, *enums* and primitive data types) is compared to the constants – if it equals a constant, the corresponding instruction is executed; if it equals none of the constants, the instruction belonging to the *default*-case is executed
- The instruction *break* ends the switch-statement – if it is left out, every single instruction after the first time the expression equals a constant is executed

Task 1: Test scores

- Write a program that receives a test score (an integer value between 0 and 100) and prints out the grade that is achieved by the given score!
  - The grade for a score of 90 or higher is „A“
  - The grade for a score of 80 or higher is „B“
  - The grade for a score of 70 or higher is „C“
  - The grade for a score of 60 or higher is „D“
  - The grade for a score of less than 60 is „F“

The *while* and the *do-while* Statement

- *while (booleanExpression) {code}*
  - If the *boolean expression* evaluates to *true*, the code in parentheses is executed – the control flow then jumps back to the evaluation of the *boolean expression*
  - As long as the *boolean expression* remains *true*, the code is executed repeatedly, until the *boolean expression* evaluates to false
- *do {code} while (booleanExpression)*
  - Similar to the *while* Statement, with the only exception that the code is executed at least once before the *boolean expression is evaluated*

The *for* Statement

- *for (initialization; booleanExpression; increment) {code}*
  - The *initialization* is executed only once
  - After that the *boolean expression* is evaluated – if it evaluates to *true*, the code in parentheses is executed
  - Afterwards, the increment is executed, and the *boolean expression* is evaluated again
  - As long as the *boolean expression* remains *true*, the code and the increment are executed repeatedly, until the *boolean expression* evaluates to false
    - ➢ Compact way to iterate over a range of values

Task 2: Prime numbers

- Write a program that prints out every prime number between 1 and 100!