Summary: Intro, Excursus, Outro

Algorithm:

-) A set of instructions for solving a specific task

-) Programs use algorithms as a step by step reference for carrying out operations.

-) Some algorithms are better suited than others depending on the task.

Structure of Java Code:

-) The basic unit is the class. A program must have at least one class.

-) Code blocks are delimited by a set of curly braces. Everything inside the code block is indented for better readability.

Memory Management:

-) Objects to which there is no reference any more are taken care of by the garbage collection.

-) In Java, primitive data types are passed by value, object data types are passed by reference.

-) When "==" is applied to object data types, references are compared, whereas equals() compares values. (Strings which are not explicitly created with "new" may point to an existing object with the same value and behave differently when compared with "==".)

Compiler/Interpreter:

-) A compiler translates the source code into a different form of code (compile-time) which is then read

and executed by the computer (run-time).

→ Pro: faster in execution

→ Con: not platform independent

→ An interpreter directly runs the source code without a previous compile step.

→ Pro: platform independent

→ Con: slower in execution

→ Java combines the compiler and the interpreter system, the code being compiled into a format called bytecode and afterwards interpreted by the Java Virtual Machine.

→ The Java Development Kit includes a compiler (javac) and the JVM. You can then run a program from the command line.

```
javac FileName.java
java FileName
```

Integrated Development Environment:

→ Package of programs to support software development

→ IDEs allow the implementing, testing, debugging and optimizing (e.g. refactoring) of source code.

→ e.g. Eclipse

Version Control:

→ History of every change made

→ Allows to go back to previous versions

→ Branching and merging, conflict resolution, source code backups

→ e.g. git

Testing:

- Static / dynamic
  - static tests: source code analysis, code reviews
  - dynamic tests: code is checked while being executed
    - black box / white box approach
- Test levels
  - Unit Test: individual units of the software are tested
  - Integration Test: checks interaction between integrated units
  - System Test: tests system as a whole
  - Acceptance Test: evaluates compliance with business requirements
- e.g. JUnit