

Course

INF1 - Computer Science 1

Version: 1 | Last Change: 27.09.2019 20:30 | Draft: 0 | Status: vom verantwortlichen Dozent freigegeben

^ General information

Long name	Computer Science 1
Approving CModule	INF1_BaMT
Responsible	Prof. Dr.-Ing. Arnulph Fuhrmann Professor Fakultät IME
Level	Bachelor
Semester in the year	winter semester
Duration	Semester
Hours in self-study	90
ECTS	6
Professors	Prof. Dr.-Ing. Arnulph Fuhrmann Professor Fakultät IME Prof. Dr.-Ing. Luigi Lo Iacono ehemaliger Professor Fakultät IME
Requirements	none
Language	German
Separate final exam	Yes

Final exam

Details

Students must demonstrate the following competences in a written examination:

- Development of short programmes to solve defined problems described in colloquial language (K.4)
- Development of short programs to solve abstractly described problems (K.2, K.5)
- Reading, understanding and, if necessary, correction of given program fragments (K.4, K.10)
- Evaluation of statements with regard to their correctness (K.12)

Minimum standard

At least 50% of the total number of points.

Exam Type

Students must demonstrate the following competences in a written examination:

- Development of short programmes to solve defined problems described in colloquial language (K.4)
- Development of short programs to solve abstractly described problems (K.2, K.5)
- Reading, understanding and, if necessary, correction of given program fragments (K.4, K.10)
- Evaluation of statements with regard to their correctness (K.12)

^ Lecture / Exercises

Learning goals

Knowledge

foundations
computer architectures
Von Neumann model
processor
memory
I/O
binary data coding
integer
characters and strings
floating point number
media data
images
audio

compiled, interpreted, hybrid languages

imperative programming
syntax, keywords, comments
variables
primitive data types
operators and expressions
arithmetic operators
boolean operators
bit operators
expressions
arithmetic
boolean
precedence of operators
elementary data structures
arrays
characters and strings
references

control flow statements

input / output

procedural programming

structuring the program code

functions

recursion

modules and libraries

modeling

object-oriented programming

classes

objects

methods

encapsulation

inheritance

polymorphism

software quality

Error handling, debugging

testing

documentation

Skills

design and modeling

abstracting problem descriptions into algorithms

deciding what programming concepts and primitives are required to solve a particular problem

design and modelling of software systems with UML

programming in Java

checking source code for programming errors

developing programs for solving concrete problems

applying fundamental programming concepts

reading and understanding third-party source code

Expenditure classroom teaching

Type	Attendance (h/Wk.)
Lecture	3
Exercises (whole course)	0
Exercises (shared course)	2
Tutorial (voluntary)	2

Separate exam

Exam Type

solving exercises within limited functional / methodical scope

Details

Independent solving of self-learning tasks on the topics of the lecture in the form of the development of more complex programs to solve problems described in colloquial or abstract language (K.4, K.5, K.9, K.2).

Minimum standard

More than 80% of all exercises submitted. A task is deemed to have been completed if it has been solved predominantly and independently.