Module title		SM Code
Lab Course Computer Science 2		**PIN2**
Module lecturer	**Faculty**	
Prof. Dr. Oliver Sterz	Electrical Engineering and Information Technology	
Module language	**Number of SWS / WSH**	**ETCS credits**
English	2 SWS / WSH	2
Teaching format		
Lab course on computer workstations		

Semester according to the study plan	
2nd semester (Bachelor)	
Attendance/classroom hours	**Additional independent study**
1 hour (submission discussion)	Preparation and follow-up work: 59 hours (at home or in CIP-Pool). Sufficient preparation and follow-up work for the IN2 submodule is a prerequisite
Type of examination / Requirements for the award of the credit points	
Practical performance assessment	

Teaching content

As part of the course, participants independently solve programming tasks based on various specified problems. The following topics are applied in practice:

Object-oriented programming and its implementation in the C++ programming language

Power series

- UML as a description language for object-oriented program designs
- Classes and objects

- Object lifecycles
- Life cycles of objects
- Inheritance and polymorphism, virtual methods
- Abstract classes and methods
- Data encapsulation / const-correctness
- Exception mechanism
- References and other new data types
- Overloading of functions and operators
- Default parameters for functions
- Implementation of data structures and algorithms in C++
- The C++ standard library
  - Container data types
  - Template mechanism
  - Iterators

Fundamental topics in software engineering

- Problem-oriented object-oriented design of applications
- Problem-oriented development and implementation of fundamental data structures
- Problem-oriented development and implementation of simple algorithms
- Design and implementation on concepts with recursion versus iteration

Learning objective: Professional competence

After successfully completing this module, students will be able to independently solve programming problems using object-oriented programming.

Participants in the course will acquire the following knowledge (1) (10%):

- basic concepts and terms of object-oriented programming
- basic knowledge of the functionality and operation of development tools
- basic knowledge of the execution model
- in-depth knowledge of C++ language elements
- in-depth understanding of the C++ memory model

- basic concepts of version management in software development

Participants in the course will acquire the following skills (2) (40%):

- independent implementation of existing algorithms in C++
- independent understanding of third-party implementations in C++ based on the source code
- independent design of simple object-oriented software solutions
- independent use of debugging tools for troubleshooting
- documentation (UML class diagrams, comments, documentation tools such as Doxygen)
- presentation of self-developed software solutions and discussion of controversial solution approaches
- creation of object-oriented software designs and their correct implementation
- working with development environments
- working with modern version management software for source code management and collaboration
- practical application of object orientation in programs
- insight into the importance of non-functional properties (maintainability, development effort, minimal redundancy in source code) and possibilities for implementation

Participants in the course will acquire the following technical and non-technical skills (3) (30%):

- independent problem analysis and structured problem-solving thinking
- independent solving of low to medium complexity problems by designing C++ programs
- independent troubleshooting and debugging of own and third-party C++ programs
- independent design of powerful, error-free, and robust C++ programs
- assessment of program performance and resource consumption
- assessment of the plausibility of program results

Literature

Recommended reading

- Kirch, U., & Prinz, P. (2022). *C++ Lernen und professionell anwenden*. Mitp-Verlag
- Pohl, I. (1999). *C++ for C Programmers*. Addison-Wesley

- Meyers, S. (2005). *Effective C++: 55 Specific Ways to Improve Your Programs and Designs*. Pearson
- Stroustrup, B. (2013). *The C++ Programming Language: Fourth Edition*. Addison-Wesley
- Wolf, J. (2013). *Grundkurs C++*. Galileo Computing
- Wolf, J. (2014). *C++: Das umfassende Handbuch*. Galileo Computing
- Lippman, S. B., Lajoie, J., & Moo, B. E. (2012). *C++ Primer*. Pearson
- Koenig, A., & Moo, B. E. (2000). *Accelerated C++: Practical Programming by Example*. Addison-Wesley
- Reese, R.M. (2013). *Understanding and Using C Pointers*. O'Reilly Media
- Free book: *"http://de.wikibooks.org/wiki/Datei:Cplusplus.pdf"*

The numbers in brackets indicate the levels to be achieved: (1)-know | (2)-can | (3)-understand and apply