Studiengang:	Bachelor Bioinformatik
Modulbezeichnung:	**Software Engineering Lab**
ggf. Kürzel:	PI-B-1
ggf. Untertitel:	
ggf. Lehrveranstaltungen:	Daily exercises and lectures (first few weeks) Daily project work with tutoring
Semester:	2. Semester
Angebotsturnus:	Lecture free time after ss
Modulverantwortliche(r):	Prof. Dr. Sven Apel
Dozent(in):	Prof. Dr. Sven Apel
Sprache:	Englisch
Zuordnung zum Curriculum:	Pflichtmodulelement der Kategorie „Praktikum der Informatik“
Lehrform / SWS:	Block (7 weeks)
Arbeitsaufwand:	270 h = 35 h of lectures and exercises + 235 h project work
Kreditpunkte:	9
Voraussetzungen:	The completed programming 2 course is an admission requirement for the Software Engineering lab
Lernziele / Kompetenzen:	Participants acquire the ability to solve complex software development problems individually and in teams. Participants are aware of common problems and pitfalls of software development and know how to address them. Participants are able to accomplish and coordinate software development tasks based on a set of given requirements. For this purpose, they are able to select proper methods and techniques to minimize risks and maximize software quality. Participants know about foundations and principles of software design, including cohesion, coupling, modularity, encapsulation, abstraction, and information hiding. They are acquainted with a whole array of design patterns, knowing their aim and individual strengths and weaknesses. They are able to apply design patterns beneficially and to judge and improve the quality of software designs. Participants master fundamental techniques and tools for software testing, debugging, and version control.
Inhalt:	• Software design • Software testing

	• Team work • Debugging
Studien- Prüfungsleistungen:	The goal of the Software Engineering Lab is to develop a non-trivial software system, partly in team effort and partly in individual effort. In this course, a number of documents (design models, documentation, etc.) and artifacts (source code, tests, etc.) need to be developed and submitted. Correctness, quality, and timely submission of all documents and artifacts are major grading criteria. The Software Engineering Lab consists of three phases: exercise, group, and individual phase. In the exercise phase, participants will complete an entry exam (mini-tests), covering current topics from the lecture. In the group phase, participants will design, implement, and test a substantial software system in a team effort. Only participants that have passed the exercise phase will be admitted to the group phase. In the individual phase, participants will design, develop, and test a smaller system (or extension to a larger system) in an individual effort. Only participants that have passed the group phase will be admitted to the individual phase. All documents (design models, documentation, etc.) and artifacts (source code, tests, etc.) of the three phases will be evaluated based on the principles and quality standard conveyed in the lectures. More details on the exams will be announced at the beginning of the course.
Literatur:	• Software Engineering. I. Sommerville, Addison-Wesley, 2004. • Software Engineering: A Practioner's Approach. R. Pressman, McGraw Hill Text, 2001. • Using UML: Software Engineering with Objects and Components. P. Stevens, et al., Addison-Wesley, 1999. • UML Distilled. M. Fowler, et al., Addison-Wesley, 2000. • Objects, Components and Frameworks with UML, D. D'Souza, et al., Addison-Wesley, 1999. • Designing Object-Oriented Software. R. Wirfs-Brock, et al., Prentice Hall, 1990. • Design Patterns. Elements of Reusable Object-Oriented Software. E. Gamma, et al., Addison-Wesley, 1995. • Head First Design Patterns. E. Freeman, et al. O'Reilly, 2004. • Software Architecture: Perspectives on an Emerging Discipline. M. Shaw, et al., Prentice-Hall, 1996. • Refactoring: Improving the Design of Existing Code. M. Fowler, et al., Addison-Wesley, 1999. • Software Testing and Analysis: Process, Principles and Techniques. M. Pezze, Wiley. 2007.