LEHRVERANSTALTUNGSBESCHREIBUNG

Überprüfung und Validierung von Softwaresystemen

Akademisches Jahr 2025 – 2026

1. Angaben zum Programm

1.1. Hochschuleinrichtung	Babes-Bolyai Universität
1.2. Fakultät	Mathematik und Informatik
1.3. Department	Informatik
1.4. Fachgebiet	Informatik
1.5. Studienform	Bachelor
1.6. Studiengang / Qualifikation	Informatik in deutscher Sprache
1.7. Form des Studiums	Vollzeit

2. Angaben zum Studienfach

2.1. LV-Bezeichnung						Code der LV	**MLG5014**
2.2. Lehrverantwortlicher – Vorlesung	Lector dr. ing. Kuderna-Iulian Bența						
2.3. Lehrverantwortlicher – Seminar	Lector dr. ing. Kuderna-Iulian Bența						
2.4. Studienjahr	3	2.5. Semester	6	2.6. Prüfungsform	E	2.7. Art der LV	Pflichtfach

3. Geschätzter Workload in Stunden

3.1. SWS	**4**	von denen: 3.2 Vorlesung	**2**	3.3. Seminar/Übung/Project	**2**
3.4. Total ore din planul de învățământ	48	von denen: 3.5 Vorlesung	24	3.6 Seminar/Übung/Project	**24**

Verteilung der Studienzeit:	**Std.**
Studium nach Handbücher, Kursbuch, Bibliographie und Mitschriften	34
Zusätzliche Vorbereitung in der Bibliothek, auf elektronischen Fachplattformen und durch Feldforschung	35
Vorbereitung von Seminaren/Übungen, Präsentationen, Referate, Portfolios und Essays	46
Tutoriat (consiliere profesională)	6
Prüfungen	6
Andere Tätigkeiten:	0

3.7. Gesamtstundenanzahl Selbststudium	**127**
3.8. Gesamtstundenanzahl / Semester	**175**
3.9. Anrechnungspunkte	**7**

4. Voraussetzungen (falls zutreffend)

4.1. zur Lehrveranstaltung	Objektorientierte Programmierung, Fortgeschrittene Programmiermethoden, Entwurfs- und Programmierumgebungen, Webprogrammierung
4.2. kompetenzbezogene	Kenntnisse der Programmierumgebungen für die Entwicklung von Anwendungen in objektorientierten Hochsprachen

5. Bedingungen (falls zutreffend)

5.1. zur Durchführung der Vorlesung	Klassenzimmer mit Videoprojektor
5.2. zur Durchführung des Seminars / der Übung	Labor

6.1. Spezifische erworbene Kompetenzen[1]

[1] Man kann Kompetenzen oder Lernergebnisse, oder beides wählen. Wenn nur eine Option ausgewählt wird, wird die Tabelle für die andere Option gelöscht, und die beibehaltene Option erhält die Nummer 6.

Berufliche/Wesentliche Kompetenzen	• Programmierung in höheren Programmiersprachen • Entwicklung und Wartung von Computeranwendungen • Einsatz von IT-Tools in einem interdisziplinären Kontext • Anwendung der theoretischen Grundlagen der Informatik und formaler Modelle
Transversale Kompetenzen	• Anwendung der Regeln für organisierte und effiziente Arbeit, verantwortungsbewusstes Verhalten im Bereich Lehre und Wissenschaft, kreative Nutzung des eigenen Potenzials unter Einhaltung der Grundsätze und Normen der Berufsethik • Effiziente Durchführung der in einer interdisziplinären Gruppe organisierten Aktivitäten und Entwicklung empathischer Fähigkeiten zur zwischenmenschlichen Kommunikation, zum Aufbau von Beziehungen und zur Zusammenarbeit mit verschiedenen Gruppen

6.2. Lernergebnisse

Kennt-nisse	Der/die Studierende kennt die Methoden zum Testen und Überprüfen von Softwaresystemen. Der/die Studierende ist mit den Instrumenten zum Testen, Debuggen und Validieren von Softwareanwendungen vertraut. Der/die Studierende ist mit Projektmanagement-Instrumenten, Versionskontrollsystemen sowie den Konzepten, Methoden und Instrumenten der kontinuierlichen Integration/kontinuierlichen Bereitstellung (CI/CD) vertraut.
Fähigkeiten	Der/die Studierende ist in der Lage , vorhandene Module und Umgebungen für die Anwendungsentwicklung auszuwählen und zu nutzen. Der/die Studierende ist in der Lage, Architekturvorlagen, Entwurfsvorlagen und bewährte Verfahren in diesem Bereich anzuwenden, um komplexe Softwareanwendungen zu entwerfen. Der/die Studierende ist in der Lage, automatisierte Tests mit unterschiedlicher Granularität zu erstellen, um die Qualität der entwickelten Systeme sicherzustellen.
Verantwortung und Autonomie	Der/die Studierende verfügt über die Fähigkeit, selbstständig zu arbeiten, um Informationen aus verschiedenen Quellen zu beobachten und zu beschaffen. Der/die Studierende verfügt über die Fähigkeit, selbstständig zu arbeiten, um komplexe Probleme zu identifizieren und damit zusammenhängende Probleme zu untersuchen, um Lösungsoptionen zu entwickeln und Lösungen umzusetzen. Der/die Studierende verfügt über die Fähigkeit, selbstständig zu arbeiten, um verschiedene Architekturen und mögliche Lösungen für ein Problem zu bewerten und diejenigen auszuwählen, die für die spezifischen Anforderungen und Einschränkungen der zu entwickelnden Anwendung geeignet sind.

7. Ziele (entsprechend der erworbenen Kompetenzen)

7.1 Allgemeine Ziele der Lehrveranstaltung	• Verständnis der Begriffe „teilweise korrekter Algorithmus" und „vollständig korrekter Algorithmus"; • Entwicklung von Fähigkeiten zum Entwerfen von Algorithmen und zum Nachweis ihrer Korrektheit; • Kenntnis der Methoden zum Testen und Überprüfen von Softwaresystemen; • Entwicklung von Fähigkeiten zum Entwerfen korrekter Programme anhand von Spezifikationen; • Entwicklung eines modernen Programmierstils.

7.2 Spezifische Ziele der Lehrveranstaltung	• Die Studierenden wissen, wie eine Inspektion abläuft und welche Schritte sie umfasst, sei es des Quellcodes oder der Spezifikation in jeder Entwicklungsphase des Softwaresystems. • Die Studierenden werden in der Lage sein, bereits in der Spezifikations- und Entwurfsphase Testfälle zu erstellen, die ihnen bei der Entwicklung eines robusteren Softwaresystems helfen. • Die Studierenden werden in der Lage sein, die Instrumente für das Testmanagement einzusetzen. • Die Studierenden werden in der Lage sein, Testfälle unter Verwendung verschiedener Kriterien (Black-Box, White-Box) zu entwerfen.

8. Inhalt

8.1 Vorlesung	Lehr-und Lernmethode	Anmerkungen
1. Softwaresystemen Verifikation und Validierung. Programm Inspektion	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
2. Programmtesten (1): das Konzept der Programm Inspektion. Testen Kriterien. White-box testen	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
3. Programmtesten (2): Testen Kriterien. White-box testen.	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
4. Levels testen. Test Typen.	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
5. Web Anwendung Testen.	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
6. Symbolische Ausführung.	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
7. Modellen Verifikation.	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
8. Die Theorie der Programmkorrektheit (I) - Korrektheit Konzept Entwicklung - Die Beiträge von Floyd, Hoare	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
9. Die Theorie der Programmkorrektheit - Die Beiträge von Dijkstra	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
10. Softwareprodukte Qualitätssicherung. Qualitätskontrolle.	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
11. Testfähigkeiten und Aufgaben des Testers	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
12. Berichte Präsentationen	Interaktive Präsentation, Erklärungen, Konversation, didaktische Demonstration	
Literatur		
8.2 Seminar / Laborarbeit	Lehr-und Lernmethode	Anmerkungen
Ü1: Inspektion L1: Inspektion Inspektionswerkzeuge. Issue-Tracker-Tools. Testmanagement-Tools	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	

Ü2: Black-box Testing (BBT) Spezifikation L2: Design von Testfällen basierend auf Spezifikationen (BBT) Inspektionswerkzeuge. Issue-Tracker-Tools. Testmanagement-Tools	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
Ü3. White-box Testing (WBT) Spezifikation. L3: Design von Testfällen basierend auf Spezifikationen (WBT) Inspektionswerkzeuge. Issue-Tracker-Tools. Testmanagement-Tools	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
U4: Test level L4:Test level Inspektionswerkzeuge. Issue-Tracker-Tools. Testmanagement-Tools	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
Ü5: Korrektheit, Floyd, Hoare L5: GBO und Web Testen Web-Test Wezeuge, Inspektionswerkzeuge. Issue-Tracker-Tools. Testmanagement-Tools	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	
Ü6: Verfeinerung von Spezifikationen L6: Statische Analyse: JML, ESC2Java	Projekte, Aufgabenlösen, Selbststudium, Gruppenübungen, Unterrichtsgespräch	

Literatur

In deutscher Sprache:

1. Kleuker, S., Formale Modelle der Softwareentwicklung, Vieweg Teubner, 2009.
2. Haubelt, C., Teich, J., Digitale Hardware/Software-Systeme, Spezifikation und Verifikation, Springer, 2010.
3. A. Spillner, M. Winter, A. Pietschker (Hrsg.), Test, Analyse und Verifikation von Software – gestern, heute, morgen, November 2017, 224 Seiten, Broschur, dpunkt.verlag, ISBN Print: 978-3-86490-470-7
4. R. Osherove, The Art of Unit Testing 2. Auflage 2015, 312 Seiten, MITP, ISBN: 9783826697128

Literatur in anderen Sprachen

1. Frentiu, M., Verificarea si validarea sistemelor soft, Presa Universitara Clujeana, 2010
2. R. S. Pressman, Software engineering: a practinioner's approach, seventh edition, Higher Education, 2010
3. L. Crispin, J. Grecory, Agile testing: a practical guide for testers and agile teams, Addison-Wesley, 2009
4. M. Pezzand, M. Young, Software Testing and Analysis: Process, Principles and Techniques, John Wiley & Sons, 2008

9. Verbindung der Inhalte mit den Erwartungen der Wissensgemeinschaft, der Berufsverbände und der für den Fachbereich repräsentativen Arbeitgeber

- Die Studenten erwerben die Möglichkeit, die Tools zur Verwaltung des Testprozesses zu verwenden
- Den Studenten werden verschiedene Testmethoden vorgestellt, die sie auf Softwareprodukte anwenden werden
- Die Studenten erwerben die Möglichkeit, Testfälle anhand verschiedener Kriterien (Black-Box, White-Box) zu entwerfen.

10. Prüfungsform

Veranstaltungsart	10.1 Evaluationskriterien	10.2 Evaluationsmethoden	10.3 Anteil an der Gesamtnote
10.4 Vorlesung	Schriftliche Prüfung (die **E** Note)	Schriftliche Prüfung	40%

	Verifikation und Validierung Berichte Präsentation	Mündliche Prüfung	Bonus (ab 0 bis 2 Punkte in die Schlussnote)
10.5 Seminar / Übung	Die Teilnahme an Diskussionen und Übungen wird vermerkt (die **S** Note)	Seminaraktivität	10%
	Die Laboraktivität wird anhand der Anforderungen notiert (die **L** Note)	Bewertung von Laborthemen	50%

Anmerkungen:
- Die Laborarbeit wird während der Prüfungs- oder Nachprüfungssessionen nicht erneut bewertet.
- Die Laborarbeit für Studierende, die die Prüfung nicht bestanden haben, muss wiederholt werden.
- Die Teilnahme an der Laborarbeit ist nur mit der Gruppe möglich, zu der der Studierende gehört.
- Eine verspätete Abgabe wird mit 2 Punkten Abzug von der Note bestraft.

Nach Ablauf der Abgabefrist wird die Laborarbeit mit 1 bewertet.

10.6 Minimale Leistungsstandards

- Praktische Fähigkeit zur Anwendung von Instrumenten für das Testmanagement
- Theoretisches Verständnis und praktische Fähigkeit zur Anwendung verschiedener Kriterien für die Gestaltung von Testfällen (Black-Box, White-Box).
- Kenntnis verschiedener Verifizierungsmethoden (Inspektion, Test, Korrektheitsnachweis).
- Teilnahmebedingungen für die Abschlussprüfung: 75 % der Seminaraktivitäten (mindestens 4 Anwesenheiten) und 90 % der Laboraktivitäten (mindestens 5 Anwesenheiten).
- Das Bestehen des Fachs setzt die Teilnahme an der Prüfung in der Prüfungssession und das Erreichen einer Durchschnittsnote von M >= 5,00 voraus, wobei M=40 % E+10 % S+50 % L+ R, gemäß den obigen Erläuterungen.

11. SDD-Nachhaltigkeits-Logos (Sustainable Development Goals)[2]

Nicht anwendbar.

Ausgefüllt am:
April 2025

Vorlesungsverantwortlicher

Lekt. dr. ing. Kuderna-Iulian Benţa

Seminarverantwortlicher

Lekt. dr. ing. Kuderna-Iulian Benţa

Genehmigt im Department am:
April 2025

Departmentleiter/in

Assoc.prof.phd. Adrian STERCA

[2] Bitte belassen Sie nur die Logos, die entsprechend den *Regularien zu Anwendung der Nachhaltigkeits-Logos im akademischen Betrieb* dem jeweiligen Studienfach entsprechen und löschen Sie diejenigen Logos, inklusive das allgemeine *Nachhaltigkeits-Logo* falls dieses nicht zutrifft. Falls keines der Logos für das Studienfach anwendbar ist, löschen Sie alle mit der Angabe „*nicht anwendbar*".