LEHRVERANSTALTUNGSBESCHREIBUNG
Objektorientierte Programmierung
Akademisches Jahr 2025-2026

1. Angaben zum Programm

1.1. Hochschuleinrichtung	Babeș-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	IF

2. Angaben zum Studienfach

2.1. LV-Bezeichnung	Objektorientierte Programmierung					Code der LV	MLG5006
2.2. Lehrverantwortlicher – Vorlesung	Ioan Crișan						
2.3. Lehrverantwortlicher – Seminar	Ioan Crișan						
2.4. Studienjahr		2.5. Semester		2.6. Prüfungsform	E	2.7. Art der LV	Pflichtfach

3. Geschätzter Workload in Stunden

3.1. SWS	**5**	von denen: 3.2 Vorlesung	**2**	3.3. Seminar/Übung	**1+2**
3.4 Gesamte Stundenanzahl im Lehrplan	70	von denen: 3.5 Vorlesung	28	3.6 Seminar/Übung	**14+28**
Verteilung der Studienzeit:					**Std.**
Studium nach Handbücher, Kursbuch, Bibliographie und Mitschriften					20
Zusätzliche Vorbereitung in der Bibliothek, auf elektronischen Fachplattformen und durch Feldforschung					20
Vorbereitung von Seminaren/Übungen, Präsentationen, Referate, Portfolios und Essays					20
Tutoriat (consiliere profesională)					14
Prüfungen					6
Andere Tätigkeiten:					-
3.7. Gesamtstundenanzahl Selbststudium	**80**				
3.8. Gesamtstundenanzahl / Semester	**150**				
3.9. Anrechnungspunkte	**6**				

4. Voraussetzungen (falls zutreffend)

4.1. zur Lehrveranstaltung	Grundlagen der Programmierung, Datenstrukturen und Algorithmen
4.2. kompetenzbezogene	Programmierkenntnisse

5. Bedingungen (falls zutreffend)

5.1. zur Durchführung der Vorlesung	Vorlesungsraum, Beamer, Laptop
5.2. zur Durchführung des Seminars / der Übung	Labor ausgestattet mit C++ und QT

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	• K1.1 Geeignete Beschreibung der Paradigmen der Programmierung und der spezifischen Sprachmechanismen, sowie die Identifizierung der Differenzen zwischen semantischen und syntaktischen Aspekten • K1.2 Eklärung existierender Softwareanwendungen auf verschidenen Niveaus (Architektur, Pakete, Klassen, Methoden), anhand geeigneter Anwendung der Grundkenntnisse • K1.3 Entwickeln von geeigneten Quellcodes und unitäres Testen von Komponenten in einer bekannten Programmiersprache, anhand gegebener Entwurfsspezifikationen • K1.4 Testen der Anwendungen anhand von Testplänen • K1.5 Entwurf von Programmeinheiten und Verfassung der geeigneten Dokumentationen
Transversale Kompetenzen	• TK1 Anwendung der Regeln für gut organisierte und effiziente Arbeit, für verantwortungsvolle Einstellungen gegenüber der Didaktik und der Wissenschaft, für kreative Förderung des eigenen Potentials, mit Rücksicht auf die Prinzipien und Normen der professionellen Ethik • TK3 Anwendung von effizienten Methoden und Techniken für Lernen, Informieren und Recherchieren, für das Entwicklen der Kapazitäten der praktischen Umsetzung der Kenntnisse, der Anpassung an die Bedürfnisse einer dynamischen Gesellschaft, der Kommunikation in rumänischer Sprache und in einer internationalen Verkehrssprache

6.2. Lernergebnisse

Kennt-nisse	Der Student kennt: • die Methoden, Algorithmen, Paradigmen und Techniken, die in verschiedenen Bereichen der Informatik verwendet werden. • die Nutzung von Computern, die Entwicklung von Programmen und Softwareanwendungen sowie die Informationsverarbeitung.
Fähigkeiten	Der Student ist in der Lage: • die Methoden, Algorithmen, Paradigmen und Techniken, die in verschiedenen Bereichen der Informatik verwendet werden, zu präsentieren und zu erklären. • Programmierparadigmen (prozedural, objektorientiert, funktional) zur Entwicklung von Softwareanwendungen zu verwenden, die den spezifischen Anforderungen des jeweiligen Fachgebiets entsprechen. • Informationen zu verstehen und effektiv zu kommunizieren.
Verantwortung und Autonomie	Der Student ist in der Lage, selbstständig zu arbeiten, um: • neue Anwendungen, Systeme oder Produkte zu entwickeln, zu entwerfen und zu erstellen, unter Anwendung bewährter Praktiken des Fachgebiets. • die Konzepte der objektorientierten Programmierung zu verstehen und anzuwenden, um Softwareanwendungen mittlerer Komplexität zu entwickeln. • Computerprogramme zu entwerfen und Softwaresysteme zu analysieren.

7. Ziele (entsprechend der erworbenen Kompetenzen)

7.1 Allgemeine Ziele der Lehrveranstaltung	• Erlernen der objektorientierter Programmierung, sowie der C++Sprache und der QT Bibliothek
7.2 Spezifische Ziele der Lehrveranstaltung	• Der Unterschied zwischen der traditionellen Programmierung und der objektorientierter Programmierung • Verstehen der Klassen als Grundstrukturen der Programmierung • Programmieren in C++ und QT

8. Inhalt

8.1 Vorlesung	Lehr-und Lernmethode	Anmerkungen

1. Objektorientierte Paradigma • Grundlagen von C • Lexikale Elemente • Datentypen, Variablen, Konstanten • Funktionen	Darstellung der Thematik, Diskussion	
2. Modulare Programmierung in C++ • Funktionen. Parameter • Header Dateien, Bibliotheken	Vortrag, Beweis, Diskussion	
3. Die C++ Programmier-sprache. Modern C++ - C++1,14,17 – C++ Core Guideline Abgeleitete Datentypen • Vektoren und Strukturen • Pointer	Vortrag, Beweis, Diskussion	
4. Objektorientierte Programmierung in C++ • Klassen und Objekte • UML Diagramme für Klassen	Vortrag, Beweis, Diskussion	
5. Generische Programmierung	Vortrag, Beweis, Diskussion	
6. Resource Management (Memory) in C++	Vortrag, Beweis, Diskussion	
7. Vererbung • Substitutionsprinzip • Abgeleitete Klassen • UML Darstellungen	Vortrag, Beweis	
8. Polymorphismus	Vortrag, Beweis, Diskussion	
9. Benutzerschnittstellen	Vortrag, Diskussion	
10. Ereignisgesteuerte Programmierung I	Vortrag, Beweis, Diskussion	
11. Ereignisgesteuerte Programmierung II	Vortrag, Beweis	
12. Die STL Bibliothek. Schablone	Vortrag, Beweis	
13. POS Anwendung	Vortrag, Beweis	
14. Wiederholung	Vortrag, Beweis, Diskussion	

Literatur

- Bruce Eckel, Thinking in C++, www.bruceeckel.com
- Alexandrescu, Programarea moderna in C++. Programare generica si modele de proiectare aplicate, Editura Teora, 2002
- M. Frentiu, B. Parv, Elaborarea programelor. Metode si tehnici moderne, Ed. Promedia, Cluj-Napoca, 1994.

Literatur in deutscher Sprache

- G. Goos, W. Zimmermann, Objektorientiertes Programmieren und Algorithmen, Springer, Berlin, Heidelberg, New York, 2006.
- Pötzsch-Heffter, A., Konzepte objektorientierter Programmierung, Springer, Berlin, Heidelberg, 2009.
- Küchlin, W, Weber, A., Einführung in die Informatik, Objektorientiertes Programmieren mit Java, Springer, Berlin, Heidelberg, New York, 2004.
- B. Stroustup, Die C++ Programmiersprache, Addison Wesley, 2000.

8.2 Seminar / Laborarbeit	Lehr-und Lernmethode	Anmerkungen
Seminar 1. Einfache Aufgaben in C, lokale und globale Variablen, Vektoren und Strukturen. Labor 1: MinGW und Eclipse CDT Installation. Spezifikation, Design und Implementierung einfacher Aufgaben in C/C++.	Beispiele, Diskussionen	
Labor 2. Modulare Programmierung in C++	Beispiele, Diskussionen	

Seminar 2. Container TAD, Darstellungen Labor 3: Feature driven development	Beispiele, Diskussionen	
Labor 4: Feature driven development	Beispiele, Diskussionen, Gruppenarbeit	
Seminar 4. Dynamische Vektoren Klassen. Iterierung Labor 5: Feature driven development	Beispiele, Diskussionen	
Labor 6: Architekturen	Beispiele, Diskussionen	
Seminar 5. Abstrakte Klassen, Polimorphismus Labor 7: Architekturen	Beispiele, Diskussionen	
Labor 8: Architekturen	Beispiele, Diskussionen	
Seminar 6. Template Klassen Labor 9: Text Dateien Labor 10: GUI mit QT	Beispiele, Diskussionen, Gruppenarbeit	
Seminar 7: Lösen komplexer Aufgaben mit UML Diagramme. Labor 11: Repository	Beispiele, Diskussionen, Gruppenarbeit	
Labor 12. Container, STL Algorithmen	Beispiele, Diskussionen	
Labor 13. Abgabe Laborarbeiten	Beispiele, Diskussionen	
Labor 14. Aufgaben: Abgabe Laborarbeiten	Beispiele, Diskussionen, Gruppenarbeit	

Literatur

- Bruce Eckel, Thinking in C++, www.bruceeckel.com
- Alexandrescu, Programarea moderna in C++. Programare generica si modele de proiectare aplicate, Editura Teora, 2002
- M. Frentiu, B. Parv, Elaborarea programelor. Metode si tehnici moderne, Ed. Promedia, Cluj-Napoca, 1994.

Literatur in deutscher Sprache:

- G. Goos, W. Zimmermann, Objektorientiertes Programmieren und Algorithmen, Springer, Berlin, Heidelberg, New York, 2006.
- Pötzsch-Heffter, A., Konzepte objektorientierter Programmierung, Springer, Berlin, Heidelberg, 2009.
- Küchlin, W, Weber, A., Einführung in die Informatik, Objektorientiertes Programmieren mit Java, Springer, Berlin, Heidelberg, New York, 2004.
- B. Stroustup, Die C++ Programmiersprache, Addison Wesley, 2000.

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

Diese Vorlesung wird an international bekannten Universitäten im Fachgebiet Informatik angeboten.
Der Inhalt der Vorlesung ist wichtig für die Softwarefirmen und entspricht der ACM Richtlinien.

10. Prüfungsform

Veranstaltungsart	10.1 Evaluationskriterien	10.2 Evaluationsmethoden	10.3 Anteil an der Gesamtnote
10.4 Vorlesung	Korrekter Umgang mit den Grundbegriffen der objektorientierter	schriftliche Abschlussarbeit	40%

	Programmierung, Fähigkeit Programme in C++ zu schreiben		
10.5 Seminar / Übung	Fähigkeit die QT Bibliothek für das Testen der C++ Programme zu benutzen Überprüfung der Korrektheit der abgegenen C++ Programme	Diskussion	30% 30%
10.6 Minimale Leistungsstandards			
• Für das Bestehen der Prüfung muss die Mindestnote 5 erzielt werden.			

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

	Allgemeines Logo für die SDG-Initiative							

Ausgefüllt am:
1.04.2025

Vorlesungsverantwortlicher

Ioan Crişan

Seminarverantwortlicher

Ioan Crişan

Genehmigt im Department am:
25.04.2025

Departmentleiter

Conf. dr. 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*".