LEHRVERANSTALTUNGSBESCHREIBUNG

1. Angaben zum Programm

1.1 Hochschuleinrichtung	**Babes-Bolyai Universität, Cluj-Napoca**
1.2 Fakultät	Mathematik und Informatik
1.3 Department	Informatik
1.4 Fachgebiet	Informatik
1.5 Studienform	Bachelor
1.6 Studiengang / Qualifikation	Informatik

2. Angaben zum Studienfach

2.1 LV-Bezeichnung	GRUNDLAGEN DER PROGRAMMIERUNG						
2.2 Lehrverantwortlicher – Vorlesung	Lect. Dr. Cătălin Rusu						
2.3 Lehrverantwortlicher – Seminar	Lect. Dr. Cătălin Rusu						
2.4 Studienjahr	1	2.5 Semester	1	2.6. Prüfungsform	Prüfung	2.7 Art der LV	Verpflichtend

3. Geschätzter Workload in Stunden

3.1 SWS	2	von denen: 3.2 Vorlesung	2	3.3 Seminar/Übung	2 Sem 2 Labor
3.4 Gesamte Stundenanzahl im Lehrplan	84	von denen: 3.5 Vorlesung	28	3.6 Seminar/Übung	56
Verteilung der Studienzeit:					Std.
Studium nach Handbücher, Kursbuch, Bibliographie und Mitschriften					14
Zusätzliche Vorbereitung in der Bibliothek, auf elektronischen Fachplattformen und durch Feldforschung					12
Vorbereitung von Seminaren/Übungen, Präsentationen, Referate, Portfolios und Essays					14
Tutorien					8
Prüfungen					18
Andere Tätigkeiten:					-

3.7 Gesamtstundenanzahl Selbststudium	66
3.8 Gesamtstundenanzahl / Semester	150
3.9 Leistungspunkte	6

4. Voraussetzungen (falls zutreffend)

4.1 curricular	•
4.2 kompetenzbezogen	•

5. Bedingungen (falls zutreffend)

5.1 zur Durchführung der Vorlesung	• Vorlesungsraum, Beamer, Laptop
5.2 zur Durchführung des Seminars / der Übung	• Laborräume mit Python ausgestattet

6. Spezifische erworbene Kompetenzen

Berufliche 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
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

7. Ziele (entsprechend der erworbenen Kompetenzen)

7.1 Allgemeine Ziele der Lehrveranstaltung	Kenntnis der grundlegenden Begriffe des Software Engineerings, sowie der Programmiersprache Python.
7.2 Spezifische Ziele der Lehrveranstaltung	• Kenntnis der grundlegenden Begriffe des Programmierens, sowie deren des Software Engineerings. • Anwendung der Programmaufbau Tools • Erlernen von Python, sowie verschiedene Platformen und Tools

8. Inhalt

8.1 Vorlesung	Lehr- und Lernmethode	Anmerkungen
1. Einleitung in die Software Entwicklung • Was is Programmieren, Grundlagen von Python, Python Interpreter, Rollen in Software Engineering • Wie schreibt man ein Programm? • Beispiele	Darstellung der Thematik, Diskussion	
2. Prozedurale Programmierung • Strukturierte Typen: Listen, Tuple, Wörterbücher • Funktionen • Parameter • Anonyme Funktionen • Wie werden Funktionen geschrieben?	Vortrag, Beweis, Diskussion	
3. Modulares Programmieren • Was ist ein Modul: Pythonmodule, Variablen Domains, Pakete, Standardmodule, Modul Verteilung • Wie organisieren wir den Sourcecode? • Eclipse+PyDev/Pycharm	Vortrag, Beweis, Diskussion	
4. User defined Typen • Wie definieren wir neue Typen? • Abstrakte Datentypen	Vortrag, Beweis, Diskussion	
5. Prinzipien der Programmierung • FDD, GRASP, DDD, Prinzipien	Vortrag, Beweis, Diskussion	
6. Objektorientierte Programmierung • Objekte und Klassen	Vortrag, Beweis, Diskussion	

• UML Diagramme		
7. Programmdesign • Top down und bottom up Strategien • UI Organisierung	Vortrag, Diskussion	
8. Programmtesten und -Inspektion	Vortrag, Beweis, Diskussion	
9. Rekursion	Vortrag, Diskussion	
10. Komplexität der Algorithmen	Vortrag, Beweis, Diskussion	
11. Backtracking	Vortrag, Diskussion	
12. Suchalgorithmen	Vortrag, Diskussion	
13. Sortierungsalgorithmen: BubbleSort, SelectionSort, InsertionSort, QuickSort, MergeSort	Vortrag, Diskussion	
14. Wiederholung	Vortrag, Beweis, Diskussion	

Literatur

1. Kent Beck.*Test Driven Development: By Example. Addison-Wesley Longman, 2002*. See also Test-driven development. http://en.wikipedia.org/wiki/Test-driven_development
2. Martin Fowler. *Refactoring. Improving the Design of Existing Code*. Addison-Wesley, 1999. See also http://refactoring.com/catalog/index.html
3. Frentiu, M., H.F. Pop, Serban G., Programming Fundamentals, Cluj University Press, 2006
4. *The Python language reference*. http://docs.python.org/py3k/reference/index.html
5. *The Python standard library*. http://docs.python.org/py3k/library/index.html
6. *The Python tutorial*. http://docs.python.org/tutorial/index.html

Auf Deutsch:

1. Allan B. Downey, Programmieren lernen mit Python, O'Reilly ISBN 978-3-86899-946-4
2. Manfred Baumgartner, Martin Klonk, Helmut Pichler, Richard Seidl, Siegfried Tanczos, Agile Testing, Hanser ISBN: 978-3-446-43194-2
3. Thomas Theis, Einstieg in Python: Ideal für Programmieranfänger

8.2 Seminar / Übung	Lehr- und Lernmethode	Anmerkungen
1. Python Programme	Beispiele, Diskussionen	
2. Prozedurale Programmierung	Beispiele, Diskussionen	
3. Modulare Programmierung	Beispiele, Diskussionen	
4. Selbstdefinierte Typen	Beispiele, Diskussionen, Gruppenarbeit	
5. Design Prinzipien	Beispiele, Diskussionen	
6. Objektorientierte Programmierung	Beispiele, Diskussionen	

7. Design	Beispiele, Diskussionen	
8. Testen und Inspektion	Beispiele, Diskussionen	
9. Rekursion	Beispiele, Diskussionen	
10. Komplexität der Algorithmen	Beispiele, Diskussionen, Gruppenarbeit	
11. Backtracking	Beispiele, Diskussionen, Gruppenarbeit	
12. Suchalgorithmen	Beispiele, Diskussionen	
13. Vorbereitung für den praktischen Test	Beispiele, Diskussionen	
14. Vorbereitung für die schriftliche Prüfung	Beispiele, Diskussionen, Gruppenarbeit	

Literatur

1. Kent Beck.*Test Driven Development: By Example. Addison-Wesley Longman, 2002*. See also Test-driven development. http://en.wikipedia.org/wiki/Test-driven_development
2. Martin Fowler. *Refactoring. Improving the Design of Existing Code*. Addison-Wesley, 1999. See also http://refactoring.com/catalog/index.html
3. Frentiu, M., H.F. Pop, Serban G., Programming Fundamentals, Cluj University Press, 2006
4. *The Python language reference*. http://docs.python.org/py3k/reference/index.html
5. *The Python standard library*. http://docs.python.org/py3k/library/index.html
6. *The Python tutorial*. http://docs.python.org/tutorial/index.html
7. *Robert Sedgewick:* Algorithmen (2. Auflage, Pearson Studium 2002)
8. Martin von Löwis, Nils Fischbeck , **Python 2 ,** Addison-Wesley-Longman (2000)
9. Tobias Himstedt and Klaus Mänzel , **Mit Python programmieren,** dpunkt.Verlag, 1999

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.

Die Vorlesung richtet sich an die IEEE und ACM Curricula Recommendations for Computer Science studies.

Der Inhalt der Vorlesung ist von Bedeutung für Software Firmen.

10. Prüfungsform

Veranstaltungsart	10.1 Evaluationskriterien	10.2 Evaluationsmethoden	10.3 Anteil an der Gesamtnote
10.4 Vorlesung	Angeeignete Kenntnisse	schriftliche Abschlussarbeit	30%
		Schriftliche Zwischenprüfung	20%
10.5 Seminar / Übung	Programmieren	Praktischer Test	20%
	Laborarbeiten	Dokumentation, Diskussion	30%

10.6 Minimale Leistungsstandards

Note jeder Übung (Labor) soll größer als 5 sein. Für das Bestehen der praktischen Prüfung muss die Mindestnote 5 erreicht werden. Für das Bestehen der schriftlichen Prüfung muss die Mindestnote 5 erzielt werden. Für das Bestehen der schriftlichen Zwischenprüfung muss die Mindestnote 5 erzielt werden.

Nur die Endnote wird auf ganze Zahl gerundet.

Erforderliche Anwesenheit beim: Seminar: 75%; Labor: 90%.

Ausgefüllt am: Vorlesungsverantwortlicher Seminarverantwortlicher

Lect. Dr. Cătălin Rusu Lect. Dr. Cătălin Rusu

Genehmigt im Department am: Departmentdirektor

Prof. Dr. Anca Andreica