**Lehrplan**

**1. Informationen zum Programm**

|  |  |
| --- | --- |
| 1.1 Hochschule | **Babeş -Bolyai-Universität** |
| 1.2 Fakultät | **Fakultät für Mathematik und Informatik** |
| 1.3 Abteilung | **Institut für Informatik** |
| 1.4 Studienfach | **Informatik** |
| 1.5 Studienzyklus | **Bachelor-Abschluss** |
| 1.6 Studiengang / Abschluss | **Informatik auf Deutsch** |

**2. Angaben zum Fach**

|  |  |
| --- | --- |
| 2.1 Bezeichnung der Disziplin ( de )( de ) | **Parallele und verteilte Programmierung** |
| 2.2 Studiengangskoordinator | **Conf. Dr. Christian Sacarea** |
| 2.3 Koordinationsseminar | **Conf. Dr. Christian Sacarea** |
| 2.4. Studienjahr | **3** | 2,5 Semester | **5** | 2.6. Prüfungsform | **C** | 2.7 Art der L V | **Wahplichtfach** |
| 2.8 Modulnummer | **MLG5165** |  |

**3. Geschätzter Gesamtaufwand** (Stunden/Semester der didaktischen Aktivitäten)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 3,1 Stunden pro Woche | 5 | Davon: 3,2 Lehrveranstaltungen | 2 | 3.3 Seminar/Labor | 3 |
| 3.4 Gesamtstundenzahl des Studienplans | 70 | Davon: 3,5 Kurse | 28 | 3.6 Seminar/Labor | 42 |
| Zeiteinteilung: | Std. |
| Lernen mit Handbuch, Kursbegleitung, Bibliographie, Vorlesungsskript | 15 |
| Zusätzliche Dokumentation (in Bibliotheken, auf elektronischen Plattformen, Felddokumentation) | 15 |
| Vorbereitung auf Seminare/Labore, Hausaufgaben, Hausarbeiten, Referate, Portfolios und Essays | 14 |
| Unterrichten | 5 |
| Bewertungen | 6 |
| Andere Aktivitäten: ................. |  |
| 3.7 Gesamtstundenzahl des individuellen Studiums | 55 |  |
| 3.8 Gesamtstundenzahl pro Semester | 125 |
| 3.9 Anzahl der ECTS-Punkte | 5 |

**4. Voraussetzungen** (falls erforderlich)

|  |  |
| --- | --- |
| 4.1. Lehrplan | * Objektorientierte Programmierung,
* Datenstrukturen und Algorithmen,
* Betriebssysteme
 |
| 4.2. Kompetenzen | * Programmierkenntnisse
 |

**5. Bedingungen** (falls erforderlich)

|  |  |
| --- | --- |
| 5.1. Natürlich | * Hörsaal mit Beamer
 |
| 5.2. für die Seminar-/Laboraktivitäten | * Labor mit Arbeitsplätzen
 |

**6. Spezifische erworbene Kompetenzen**

|  |  |
| --- | --- |
| **Fachkompetenzen** | C3.1 Identifizierung von für Informationssysteme charakteristischen Problemklassen und Lösungsmethoden  C3.2 Interdisziplinäres Wissen, Lösungsmuster und Werkzeuge nutzen, Experimente durchführen und deren Ergebnisse interpretierenC4.2 Erklären Sie die Rolle, Interaktion und Funktionsweise von Softwaresystemkomponenten   C4.5 Entwicklung, Implementierung und Integration von Softwarelösungen  |
| **Querschnittskompetenzen** | CT1 Ehrenhaftes, verantwortungsvolles, ethisches Verhalten im Sinne des Gesetzes, um den beruflichen Ruf zu wahrenCT3 Initiative und proaktives Verhalten zur Aktualisierung des beruflichen, wirtschaftlichen und organisationalen kulturellen Wissens zeigen |

**7. Ziele der Disziplin** (Ergebnisse der erworbenen Kompetenzen)

|  |  |
| --- | --- |
| 7.1 Allgemeines Ziel der Disziplin | * Erlernen Sie die wichtigsten Konzepte der gleichzeitigen, parallelen und verteilten Programmierung.
* Grundlagen der Kommunikation zwischen Prozessen und Threads, auf derselben Maschine oder auf verschiedenen Maschinen;
* Kenntnis der grundlegenden Techniken der parallelen Programmierung;
* Parallele Anwendungsentwurfsmuster kennen und nutzen
* Kennen und Verwenden vorhandener Frameworks zur Entwicklung paralleler und verteilter Anwendungen
 |
| 7.2 Spezifisches Ziel der Disziplin | * Parallele Architekturen und parallele Programmiersysteme
* Erfahren Sie, wie Sie parallele Programmiertechniken zur Problemlösung einsetzen
* Erfahren Sie, wie Sie die durch Parallelisierung erzielte Leistungssteigerung bewerten können.
* Fähigkeit, selbstständig oder im Team zu arbeiten, um Probleme in einem parallelen und/oder verteilten Kontext zu lösen
 |

**8. Inhalte**

|  |  |  |
| --- | --- | --- |
| 8.1 Verlauf | Lehr- und Lernmethode | Hinweise |
| 1. Allgemeine Einführung: Notwendigkeit der Verwendung von Parallelität; Konkurrent vs. parallel vs. verteiltes Rechnen; Ebenen der Parallelität
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Parallele Architekturen: Pipeline; Vektormaschinen; Grid- und Cluster-Computer; Supercomputer
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Prozesse vs. Threads; Verwalten von Prozessen/Threads
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Konzepte zur Parallelität: Race Conditions, kritische Abschnitte, gegenseitiger Ausschluss, Deadlock; Synchronisierungen: Monitore, Semaphoren
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Modelle der Parallelität: Implizit vs. explizite Parallelität; Datenparallelität; Nachrichtenübermittlung; gemeinsam genutzter Speicher
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Parallele Programmierung im Shared Memory: C pthreads , C++-Threads, Java-Threads, OpenMP
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Leistungsbewertung für parallele Programme:

PRAM (Parallel Random Access Machine).Effizienz, Kosten, Skalierbarkeit. | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Parallele Programmiermuster: Master-Slaves; Aufgabenfarm / Arbeitspool; teile und herrsche ; Pipeline
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Parallele Programme zur Nachrichtenübermittlung. Lampen
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Phasen in einem parallelen Programm: PCAM (Partition, Communication, Aggregation, Mapping): Aufgabenzerlegung; Domänenzerlegung (geometrische Zerlegung); Granularität; Grad der Parallelität; Aufgabenabhängigkeit
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Techniken zum Erstellen paralleler Programme: Teile und herrsche ; binärer Baum; rekursives Doubleback
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Datenparallele Programmierung
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. GPGPU (General Processing on the Graphical Verarbeitungseinheit: OpenCL, CUDA
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| 1. Verteilte Dateisysteme
 | Ausstellung, Konzepte,Beispiele, Fallstudien. |  |
| Bibliographiehttp://www.cs.ubbcluj.ro/~rlupsa/edu/pdp/1. Ian Foster. Entwerfen und Erstellen paralleler Programme, Addison-Wesley 1995.2. Michael McCool, Arch Robinson, James Reinders, Strukturierte Parallele Programmierung: Muster für„Effiziente Berechnung“, Morgan Kaufmann, 2012.3. Berna L. Massingill, Timothy G. Mattson und Beverly A. Sanders, Addison Eine Mustersprache fürParallele Programmierung. Wesley Software Patterns Series, 2004.4. Grama, A. Gupta, G. Karypis , V. Kumar. Einführung in das Parallelrechnen, Addison Wesley, 2003.5. D. Grigoras. Paralleles Rechnen. Von der System- zur Anwendungsprogrammierung. Computer Libris Agora, 2000.6. V. Niculescu. Paralleles Rechnen. Entwurf und formale Entwicklung paralleler Programme. Universitätsverlag.Cluj-Napoca , 2006.7. DB Skillicorn, D. Talia. Modelle und Sprachen für parallele Berechnungen. ACM Computerumfragen,30(2) S. 123-136, Juni 1998.8. B. Wilkinson, M. Allen, Parallele Programmiertechniken und Anwendungen mit vernetztenWorkstations und Parallelcomputer, Prentice Hall, 20029. E. F. Van de Velde. Gleichzeitiges wissenschaftliches Rechnen. Spring-Verlag, New-York Inc. 1994.10. Boian FM Ferdean CM, Boian RF, Dragos RC Parallele Programmierung auf Unix-Plattformen,Windows, Java. Hrsg. Blue, Mikroinformatik-Gruppe, Cluj, 2002.11. OpenMP-Dokumente: https://www.openmp.org/12. MPI-Dokumente: https://docs.open-mpi.org/13. CUDA-Dokumente: <https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html>14. OpenCL-Dokumente: https://www.khronos.org/opencl/ |
| 8.2 Seminar/Labor | Lehr- und Lernmethode | Hinweise |
| L1-Threads vs. Prozesse |  |  |
| L2-L5 Parallele Programmierung C++, Java, C# |  |  |
| L6-L7 OpenMP |  |  |
| L8-L10 MPI |  |  |
| L11-L14 CUDA/OpenCL |  |  |
| Bibliographie1. Eckel, B., Thinking in Java, 4. Auflage, New York: Prentice Hall, 2006.2. Larman, C.: Anwendung von UML und Entwurfsmustern: Eine Einführung in OO-Analyse und -Design, Berlin: Prentice Hall, 2004.3. Fowler, M., Muster der Unternehmensanwendungsarchitektur, Addison-Wesley, 2002.4. E. Gamma, R. Helm, R. Johnson, J. Vlissides , Design Patterns – Elemente wiederverwendbarer objektorientierter Software, Hrsg. Thomas B., 1994.5. Walls, Craig, Spring in Action, Dritte Ausgabe, Hrsg. O’Reilley , 2011.6. Kent Beck, Test Driven Development: Am Beispiel, Hrsg. Addison-Wesley Professional, 2002.7. \*\*\*, http://download.oracle.com/javase/tutorial/8. \*\*\*, http://msdn.microsoft.com/en-us/library/aa288436%28v=vs.71%29.aspx |

**9. Übereinstimmung der Inhalte der Disziplin mit den Erwartungen der Wissensgemeinschaft, der Berufsverbände und der repräsentativen Arbeitgeber im Bereich des Programms**

|  |
| --- |
| * Der Kurs folgt den Empfehlungen von ACM und IEEE für Informatikstudien
* Der Kurs ist Teil des Lehrplans aller großen Universitäten im In- und Ausland.
* Die Kursinhalte erachten die Softwareunternehmen als wichtig für den Erwerb fortgeschrittener Programmierkenntnisse.
 |

**10. Auswertung**

|  |  |  |  |
| --- | --- | --- | --- |
| Art der Tätigkeit | 10.1 Bewertungskriterien | 10.2 Bewertungsmethoden | 10.3 Anteil an der Note (%) |
| 10.4 Verlauf | Grundlegende Konzepte kennen | Schriftliche Prüfung | 40 % |
| Anwendung theoretischerWissen im ProblemLösung | Projekt | 30 % |
| 10.5 Seminar-/Laboraktivitäten | Anwendung theoretischerWissen im ProblemLösung | LaborauswertungAufgaben | 30 % |
|  |  |  |
| 10.6 Mindestleistungsstandards |
| * Mindestens 4,5 von 10 Punkten in der schriftlichen Prüfung
* Mindestens 4,5 von 10 Punkten für den Durchschnitt
 |

Datum Unterschrift Lehrveranstaltungsleiter Unterschrift Seminarleiter

13.04.2024 Conf. Dr. Christian Săcărea Conf. Dr. Christian Săcărea

Datum der Genehmigung Unterschrift des Abteilungsleiters

........................................... Conf. Dr. Adrian Sterca