

FIȘA DISCIPLINEI

1. Date despre program

1.1 Instituția de învățământ superior	Universitatea Babeș-Bolyai Cluj-Napoca
1.2 Facultatea	Facultatea de Matematica și Informatică
1.3 Departamentul	Departamentul de matematică
1.4 Domeniul de studii	Informatică
1.5 Ciclul de studii	Licenta
1.6 Programul de studiu / Calificarea	Informatică română

2. Date despre disciplină

2.1 Denumirea disciplinei	Programare paralelă și distribuită						
2.2 Titularul activităților de curs	Conf. Dr. Niculescu Virginia						
2.3 Titularul activităților de seminar	Conf. Dr. Niculescu Virginia						
2.4 Anul de studiu	3	2.5 Semestrul	5	2.6. Tipul de evaluare	E	2.7 Regimul disciplinei	Obligativ

3. Timpul total estimat (ore pe semestru al activităților didactice)

3.1 Număr de ore pe săptămână	4	Din care: 3.2 curs	2	3.3 seminar/laborator	2
3.4 Total ore din planul de învățământ	56	Din care: 3.5 curs	28	3.6 seminar/laborator	28
Distribuția fondului de timp:					ore
Studiul după manual, suport de curs, bibliografie și notițe					15
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					15
Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri					28
Tutoriat					5
Examinări					6
Alte activități:					-
3.7 Total ore studiu individual	58				
3.8 Total ore pe semestru	125				
3.9 Numărul de credite	5				

4. Precondiții (acolo unde este cazul)

4.1 de curriculum	<ul style="list-style-type: none"> Fundamentele programării, Programare orientată obiect Structuri de date și algoritmi
4.2 de competențe	<ul style="list-style-type: none"> Abilități de programare Java și C++

5. Condiții (acolo unde este cazul)

5.1 De desfășurare a cursului	<ul style="list-style-type: none"> Sala cu proiector
5.2 De desfășurare a seminarului/laboratorului	<ul style="list-style-type: none"> laborator cu stații de lucru

6. Competențele specifice acumulate

Competențe profesionale	<p>Fiecare student trebuie să dovedească faptul că a dobândit un nivel de cunoștințe și înțelegere a domeniului și că este capabil (a) să exprime aceste cunoștințe; și, de asemenea, că poate folosi aceste cunoștințe în rezolvarea unor probleme prin implementarea soluțiilor folosind programarea paralelă și distribuită.</p> <p>C6:</p> <p>C6.1 Identificarea conceptelor și modelelor de bază pentru sisteme de calcul și rețele de calculatoare.</p> <p>C6.4 Efectuarea de măsurători de performanță pentru timpi de răspuns, consum de resurse.</p> <p>C6.5 Realizarea unor proiecte de rețele de calculatoare/arhitecturi paralele și distribuite</p>
--------------------------------	---

Competențe transversale	<p>CT1 Aplicarea regulilor de muncă organizată și eficientă, a unor atitudini responsabile față de domeniul didactic-științific, pentru valorificarea creativă a propriului potențial, cu respectarea principiilor și a normelor de etică profesională</p> <p>CT3 Utilizarea unor metode și tehnici eficiente de învățare, informare, cercetare și dezvoltare a capacităților de valorificare a cunoștințelor, de adaptare la cerințele unei societăți dinamice și de comunicare în limba română și într-o limbă de circulație internațională</p>
--------------------------------	---

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	<ul style="list-style-type: none"> - Inșurirea principalelor entitati si concepte cu care se opereaza în contextul programării paralele, concurente și distribuite. - Prezentarea bazelor comunicării între procese și threaduri, aflate pe aceeași mașină sau pe mașini aflate la distanță. - Inșurirea bazelor specifice ale programării paralele, concurente și distribuite - Cunoasterea, intelegerea paradigmelor si tehnicilor de baza ale programarii paralele. - Intelegerea si folosirea unor sabloane de proiectare pentru dezvoltarea aplicatiilor paralele. - Deprinderea folosirii unor frameworkuri pentru dezvoltarea aplicatiilor paralele si distribuite.
7.2 Obiectivele specifice	<ul style="list-style-type: none"> • Arhitecturi paralele și sisteme de programare paralelă • Abilitatea de a aplica tehnici specifice programarii paralele in rezolvarea problemelor • Abilitatea de evalua cresterea de performanta obtinuta prin folosirea paralelizarii. • Abilitatea de a lucra independent sau in echipa pentru a rezolva probleme într-un programare paralela si/sau distribuita. •

8. Conținuturi

8.1 Curs	Metode de predare	Observații
C1 Introducere generala in programarea paralela si distribuita - necesitatea folosirii paralelismului; - programare paralela vs. progr. distribuita, vs. Progr. Concurenta - niveluri de folosire a paralelismului	Expunere	
C2. Arhitecturi paralele – <i>Taxonomii</i> <input type="checkbox"/> Pipeline <input type="checkbox"/> Mașini vectoriale <input type="checkbox"/> Sisteme grid și clustere <input type="checkbox"/> Supercalculatoare	Expuneri: concepte, exemple, studii de caz	
C3. Tipuri si modele de paralelism • Paralelism implicit vs. Paralelism explicit - Modelul Data-parallel - Modelul Message-passing (distributed memory) Modelul Shared-memory Procese versus fire de executie • gestiunea proceselor	Expuneri: concepte, exemple, studii de caz	
C4. Message Passing parallel programming Programare paralela in medii cu memorie distribuita. <i>MPI</i>	Expuneri: concepte, exemple, studii de caz	
C5. Programare paralela in medii cu memorie partajata - C++ Threads, Java Threads	Expuneri: concepte, exemple, studii de caz	
C6. Concurenta – concepte Race-conditions, critical section, mutual exclusion, deadlock	Expuneri: concepte, exemple, studii de caz	
C7. Concurenta – concepte Sincronizare: monitoare, semafoare, variabile conditionale	Expuneri: concepte, exemple, studii de caz	
C8. Programare paralela in medii cu memorie partajata. - OpenMP.	Expuneri: concepte, exemple, studii de caz	
C9. Evaluarea performantei programelor paralele: complexitate-timp, complexitate-procesor, acceleratie, eficienta, cost; evaluare scalabilitate	Expuneri: concepte, exemple, studii de caz	
C10 <i>Data parallel programming</i> : Cadrul general de dezvoltare a aplicațiilor GPU <input type="checkbox"/> Arhitectură; platforma NVIDIA <input type="checkbox"/> API de programare; modelul CUDA	Expuneri: concepte, exemple, studii de caz	

C11. . Faze in dezvoltarea programelor paralele (PCAM) - Partitionare, Comunicare, Aglomerare, Mapare Partitionare->Descompunere - functionala (task decomposition) - a domeniului de date(geometrica) distributii de date Granularitate, Grad de paralelizare(DOP), Task dependency	Expuneri: concepte, exemple, studii de caz	
C12- Sabloane pentru programarea paralela <i>Master-slaves; Task-Farm/Work-Pool; Divide & Impera; Pipeline</i>	Expuneri: concepte, exemple, studii de caz	
C13 Sabloane pentru programarea distribuita	Expuneri: concepte, exemple, studii de caz	
C14. Analiza comparativa generala a noilor concepte/tehnici/principii/sabloane introduse	studii de caz	

<http://www.cs.ubbcluj.ro/~vniculescu/didactic/>

Bibliografie

1. Ian Foster. Designing and Building Parallel Programs, Addison-Wesley 1995.
 2. Michael McCool, Arch Robison, James Reinders, Structured Parallel Programming: Patterns for Efficient Computation, Morgan Kaufmann, 2012 .
 3. [F. Buschmann](#), [K. Henney](#), [D. C. Schmidt](#). Pattern-Oriented Software Architecture Volume 4: A Pattern Language for
 4. Grama, A. Gupta, G. Karypis, V. Kumar. Introduction to Parallel Computing, Addison Wesley, 2003.
 5. D. Grigoras. Calculul Paralel. De la sisteme la programarea aplicatiilor. Computer Libris Agora, 2000.
 6. B. L. Massingill, T.G. Mattson, and B. A. Sanders, A Pattern Language for Parallel Programming. Wesley Software Patterns Series, 2004.
 7. V. Niculescu. Calcul Paralel. Proiectare si dezvoltare formala a programelor paralele. Presa Univ. Clujana, 2006.
 8. D.B. Skillicorn, D. Talia. Models and Languages for Parallel Computation. ACM Computer Surveys, 30(2) pg.123-136, June 1998.
 9. Distributed Computing Volume 4, Wiley. 2007.
 10. M. Richards. Software Architecture Patterns. Understanding Common Architecture. Patterns and When to Use Them 2015 O'Reilly Media.
 11. [D. Schmidt](#) (Author), [M. Stal](#) (Author), [H. Rohnert](#) (Author), [F. Buschmann](#). Pattern-Oriented Software Architecture Volume 2: Patterns for Concurrent and Networked Objects Volume 2. Wiley, 2000.
 12. B. Wilkinson, M. Allen, Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers, Prentice Hall, 2002
 13. E.F. Van de Velde. Concurrent Scientific Computing. Spring-Verlag, New-York Inc. 1994.
 14. Boian F.M. Ferdean C.M., Boian R.F., Dragos R.C. Programare concurenta pe platforme Unix, Windows, Java. Ed. Albastra, grupul Microinformatica, Cluj, 2002.
1. ***, Tutoriale OpenMP
 2. ***, Tutoriale MPI
 3. ***, Tutoriale CUDA

8.2 Laborator		
L1 –L2 . Threads vs. Processes	Discutii, exemplificare, evaluare	
L2-L4 MPI - exemple	Discutii, exemplificare, evaluare	
L5. Test1 practic – programare distribuita MPI	Discutii, exemplificare, evaluare	
L6- L8. Concurenta – conditionari, sincronizari cu exemplificari C, Java	Evaluare	
L9. Probleme de sincronizare	Discutii, exemplificare, evaluare	
L10. Test2 practic – programare concurenta- multithreading	Discutii, exemplificare, evaluare	
L11. Client-Server	Discutii, exemplificare, evaluare	
L12 OpenMP	Discutii, exemplificare, evaluare	
L13-L14 CUDA	Evaluare	

Bibliografie aditionala

1. Eckel, B., Thinking in Java, 4th Edition, New York: Prentice Hall, 2006.
2. Larman, C.: Applying UML and Design Patterns: An Introduction to OO Analysis and Design, Berlin: Prentice Hall, 2004.
3. Fowler, M., Patterns of Enterprise Application Architecture, Addison-Wesley, 2002.

4. E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns – Elements of Reusable Object Oriented Software, Ed. Addison Wesley, 1994.
5. J. Sanders, E. Kandrot. CUDA by Example. An Introduction to General-Purpose GPU Programming. Addison-Wesley. 2010.
6. A. WILLIAMS. C++ Concurrency in Action. PRACTICAL MULTITHREADING. MANNING, 2012.
7. ***, Tutoriale Java <http://download.oracle.com/javase/tutorial/>
8. ***, Tutoriale C# <http://msdn.microsoft.com/en-us/library/aa288436%28v=vs.71%29.aspx>
9. ***, Tutorial C++11
10. ***, OpenMP[<http://openmp.org/wp/>]
11. ***, MPI[<http://www.mpi-forum.org/>]

9. Coroborarea conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatori reprezentativi din domeniul aferent programului

- Cursul respecta recomandările curriculei ACM și IEEE pentru studiile de informatică.
- Cursul apare în planurile de învățământ a celor mai importante universități din țară și străinătate.
- Firmele de soft consideră conținutul cursului important pentru dobândirea unor abilități avansate de programare.

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 metode de evaluare	10.3 Pondere din nota finală
10.1 Curs	Cunoașterea conceptelor de baza	Examen scris	50%
10.2 Laborator	Folosirea conceptelor introduse la curs pentru rezolvarea unor probleme concrete	Programe de laborator - Evaluarea calitatii și completitudinii temelor primite	30%
		Teste practice	20%
10.4 Standard minim de performanță			
<ul style="list-style-type: none"> ○ Media finală trebuie să fie cel puțin 5 (pe o scară de la 1 la 10). ○ Nota de la examenul scris cel puțin 4.5 ○ Nota de la laborator cel puțin 4.5 			

Data completării

.....

Titular de curs

Conf. Dr. Niculescu Virginia

Titular de seminar

Conf. Dr. Niculescu Virginia

Data avizării în departament

.....

Director de departament

.....