

FIȘA DISCIPLINEI

Programare paralelă și distribuită

Anul universitar 2025-2026

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea Babes-Bolyai Cluj-Napoca
1.2. Facultatea	Facultatea de Matematica si Informatica
1.3. Departamentul	Departamentul de Informatica
1.4. Domeniul de studii	Informatica
1.5. Ciclul de studii	Licenta
1.6. Programul de studii / Calificarea	Informatica
1.7. Forma de învățământ	Cu frecventa

2. Date despre disciplină

2.1. Denumirea disciplinei		Programare paralelă și distribuită				Codul disciplinei		MLR5077		
2.2. Titularul activităților de curs			Conf. univ.dr. Niculescu Virginia							
2.3. Titularul activităților de seminar			Conf. univ.dr. Niculescu Virginia							
2.4. Anul de studiu		3	2.5. Semestrul	5	2.6. Tipul de evaluare		E	2.7. Regimul disciplinei		Obligativu

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

3.1. Număr de ore pe săptămână	5	din care: 3.2. curs	2	3.3. seminar/ laborator/proiect	3
3.4. Total ore din planul de învățământ	70	din care: 3.5. curs	28	3.6 seminar/laborator/proiect	42
Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)					ore
Studiul după manual, suport de curs, bibliografie și notițe (AI)					5
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					10
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					25
Tutoriat (consiliere profesională)					5
Examinări					5
Alte activități [comunicare bidirecțională cu titularul de disciplină / tutorele]					5
3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)				55	
3.8. Total ore pe semestru				125	
3.9. Numărul de credite				5	

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	Programare orientata obiect Metode avansate de programare Medii de proiectare si programare	
4.2. de competențe	Abilitati de programare Java si C++	

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sala cu proiector	
5.2. de desfășurare a seminarului/ laboratorului	laborator cu statii de lucru	

6.1. Competențele specifice acumulate¹

Competențe profesionale /esențiale	• programarea în limbaje de nivel înalt • dezvoltarea și întreținerea aplicațiilor informatice • utilizarea instrumentelor informatice în context interdisciplinar • utilizarea bazelor teoretice ale informaticii și a modelelor formale
Competențe transversale	• 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 • desfășurarea eficientă a activităților organizate într-un grup interdisciplinar și dezvoltarea capacităților empatice de comunicare interpersonală, de relaționare și colaborare cu grupuri diverse • 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

6.2. Rezultatele învățării

Cunoștințe	• Absolventul are cunoștințe necesare pentru utilizarea calculatoarelor, dezvoltarea programelor și aplicațiilor software, procesarea informațiilor. • Absolventul are cunoștințele necesare pentru selectarea și utilizarea procedurilor potrivite de instruire pentru a facilita procesul de asimilare a cunoștințelor.
Aptitudini	• Absolventul are abilitatea de a dezvolta, proiecta și crea noi aplicații, sisteme sau produse folosind bunele practici din domeniu. • Absolventul are aptitudinile necesare pentru conceperea programelor de calculator și analiza sistemelor software. • Absolventul are abilitatea de a alege și folosi module și medii existente pentru dezvoltarea de aplicațiilor.
Responsabilități și autonomie	• Absolventul poate utiliza cunoștințele și abilitățile achiziționate pentru proiectarea, organizarea și realizarea demersurilor educaționale. Absolventul este capabil de a defini/identifica/înțelege probleme de cercetare în domeniul informaticii.

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	- Însușirea principalelor concepte și mecanisme cu care se operează în contextul programării paralele, concurente și distribuite. - Prezentarea bazelor intercomunicării între procese și threaduri, aflate pe aceeași mașină sau pe mașini aflate la distanță. - Însușirea de cunoștințe de bază ale programării paralele, concurente și distribuite - Cunoașterea, înțelegerea și folosirea paradigmatelor și tehnicilor de bază ale programării paralele. - Deprinderea folosirii unor frameworkuri/APIs pentru dezvoltarea aplicațiilor paralele și distribuite.
7.2 Obiectivele specifice	• Cunoașterea elementelor de bază ale arhitecturilor paralele și ale sisteme de programare paralelă • Abilitatea de a aplica tehnici specifice programării paralele în rezolvarea problemelor. • Abilitatea de a evalua creșterea de performanță obținută prin folosirea paralelizării. • Abilitatea de a lucra independent sau în echipă pentru a rezolva probleme într-un context de programare paralelă și/sau distribuită.

¹ Se poate opta pentru competențe sau pentru rezultatele învățării, respectiv pentru ambele. În cazul în care se alege o singură variantă, se va șterge tabelul aferent celeilalte opțiuni, iar opțiunea păstrată va fi numerotată cu 6.

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 (Flynn)</i> SMP, MPP, interconnection networks Mașini vectoriale Sisteme grid și clustere, Supercalculatoare, acceleratoare	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 asincrona	Expuneri: concepte, exemple, studii de caz	
C9. Programare paralela in medii cu memorie partajata. - OpenMP	Expuneri: concepte, exemple, studii de caz	
C10 . Evaluarea performantei programelor paralele: complexitate-timp, complexitate-procesor, acceleratie, eficienta, cost; granularitate, scalabilitate	Expuneri: concepte, exemple, studii de caz	
C11. <i>Data parallel programming</i> : Cadru general de dezvoltare a aplicațiilor GPU Arhitectură; platforma NVIDIA API de programare; modelul CUDA	Expuneri: concepte, exemple, studii de caz	
C12- Sabloane de proiectare pentru programarea paralela <i>Master-slaves; Task-Farm/Work-Pool; Divide & Impera; Pipeline</i>	Expuneri: concepte, exemple, studii de caz	
C13 Sabloane de proiectare pentru programarea distribuita	Expuneri: concepte, exemple, studii de caz	
C14. Analiza comparativa generala a noilor concepte/tehnici/principii/sabloane introduse	studii de caz	
Bibliografie 1. Ian Foster. Designing and Building Parallel Programs, Addison-Wesley 1995. 2. Michael McCool, Arch Robinson, 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. M. Richards. Software Architecture Patterns. Understanding Common Architecture. Patterns and When to Use Them 2015 O'Reilly Media. 9. 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. 10. B. Wilkinson, M. Allen, Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers, Prentice Hall, 2002		

***, Tutoriale OpenMP ***, Tutoriale MPI ***, Tutoriale CUDA		
8.2 Seminar / laborator	Metode de predare	Observații
L1 –L2 . Threads vs. Processes	Discutii, exemplificare, evaluare	
L2-L4 MPI - exemple	Discutii, exemplificare, evaluare	
L5. Programare distribuita MPI	Discutii, exemplificare, evaluare	
L6- L8. Concurența – conditionari, sincronizari cu exemplificari C, Java	Discutii, exemplificare, evaluare	
L9. Probleme de sincronizare	Discutii, exemplificare, evaluare	
L10. Aplicații folosind programare concurența- multithreading	Discutii, exemplificare, evaluare	
L11- L12 Proiect Client-Server	Discutii, exemplificare, evaluare	
L13-L14 Proiect CUDA	Evaluare	
Bibliografie 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. J. Sanders, E. Kandrot. CUDA by Example. An Introduction to General-Purpose GPU Programming. Addison-Wesley. 2010. 5. A. WILLIAMS. C++ Concurrency in Action. PRACTICAL MULTITHREADING. MANNING, 2012. 6. ***, Tutoriale Java http://download.oracle.com/javase/tutorial/ 7. ***, Tutorial C++ [https://en.cppreference.com/] 8. ***, OpenMP[http://openmp.org/wp/] 9. ***, 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
10.4 Curs	Cunoașterea conceptelor și mecanismelor de bază	Examen teoretic	40%
10.5 Seminar/laborator	Folosirea conceptelor și mecanismelor introduse la curs pentru rezolvarea unor probleme concrete	Programe de laborator - Evaluarea calitatii și completitudinii temelor primite -Evaluarea înțelegerii aplicării conceptelor și mecanismelor folosite	30%
		Teste practice	30%
10.6 Standard minim de performanță			
○ Media finală trebuie să fie cel puțin 5 (pe o scară de la 1 la 10). ○ Promovarea este condiționată de nota minimă 5 la activitatea de laborator și examenul teoretic.			

11. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)²

² Păstrați doar etichetele care, în conformitate cu [Procedura de aplicare a etichetelor ODD în procesul academic](#), se potrivesc disciplinei și ștergeți-le pe celelalte, inclusiv eticheta generală pentru *Dezvoltare durabilă* - dacă nu se aplică. Dacă nicio etichetă nu descrie disciplina, ștergeți-le pe toate și scrieți "Nu se aplică".

Nu se aplică.

Data completării:
15.04.2025

Semnătura titularului de curs

..... 

Semnătura titularului de seminar

..... 

Data avizării în departament:
...

Semnătura directorului de departament

Conf.dr. Adrian STERCA