

FIȘA DISCIPLINEI

(licență)

1. Date despre program

Instituția de învățământ superior	Universitatea „Ștefan cel Mare” din Suceava
Facultatea	Facultatea de Inginerie Electrică și Știința Calculatoarelor
Departamentul	Departamentul de Calculatoare
Domeniul de studii	Calculatoare și Tehnologia Informației
Ciclul de studii	Licență
Programul de studii	Calculatoare

2. Date despre disciplină

Denumirea disciplinei	PROIECTAREA ALGORITMILOR				
Titularul activităților de curs	Prof. univ. dr. ing. Radu-Daniel VATAVU				
Titularul activităților aplicative	Prof. univ. dr. ing. Radu-Daniel VATAVU				
Anul de studiu	3	Semestrul	6	Tipul de evaluare	Examen
Regimul disciplinei	Categorია formativă a disciplinei DF - fundamentală, DD - în domeniu, DS - de specialitate, DC – complementară				DD
	Categorია de opționalitate a disciplinei: DI - impusă, DO - opțională, DF - facultativă				DI

3. Timpul total estimat (ore alocate activităților didactice)

I a) Număr de ore pe săptămână	5	Curs	3	Seminar	-	Laborator	2	Proiect	-
I b) Totalul de ore pe semestru din planul de învățământ	70	Curs	42	Seminar	-	Laborator	28	Proiect	-

II Distribuția fondului de timp pe semestru:	ore
II a) Studiul după manual, suport de curs, bibliografie și notițe	12
II b) Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren	12
II c) Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri	28
II d) Tutoriat	-
III Examinări	3
IV Alte activități (precizați):	-

Total ore studiu individual II (a+b+c+d)	52
Total ore pe semestru (Ib+II+III+IV)	125
Numărul de credite	5

4. Precondiții (acolo unde este cazul)

Curriculum	-
Competențe	-

5. Condiții (acolo unde este cazul)

Desfășurare a cursului	Laptop/PC, slide-uri suport pentru curs, exemple de programe funcționale pe calculator, conexiune la Internet	
Desfășurare aplicații	Seminar	-
	Laborator	Unități laptop/PC având instalate mediul de dezvoltare Visual Studio (Community, Code, etc.), ghid de lucrări practice în format electronic, conexiune la Internet
	Proiect	-

6. Competențe specifice acumulate

Competențe profesionale	CP1. Operarea cu fundamente științifice, ingineresti și ale informaticii CP2. Proiectarea componentelor hardware, software și de comunicații CP3. Îmbunătățirea performanțelor sistemelor hardware, software și de comunicații CP6. Proiectarea sistemelor inteligente
Competențe transversale	-

7. Obiectivele disciplinei (reieșind din grila competențelor specifice acumulate)

Obiectivul general al disciplinei	Introducere în proiectarea și analiza algoritmilor prin prezentarea, exemplificarea, discutarea și analiza comparativă a diverselor metode de proiectare a algoritmilor, evaluare a complexității teoretice și implementare de soluții algoritmice pentru probleme specifice.
	<ul style="list-style-type: none"> - Însușirea de metode de proiectare a algoritmilor (<i>e.g.</i>, Divide et Impera, programarea dinamică, etc.) ca fundamente științifice, ingineresti și ale informaticii - Însușirea metodelor de proiectare a algoritmilor pentru prelucrarea datelor în diverse reprezentări (<i>e.g.</i>, grafuri, date geometrice) și diverse aplicații (<i>e.g.</i>, sisteme inteligente) - Însușirea metodelor de analiză a complexității algoritmilor în vederea evaluării și îmbunătățirii performanțelor sistemelor software.

8. Conținuturi

Curs	Nr. ore	Metode de predare	Observații
1. Modele de calcul și analiză a complexității algoritmilor 1.1. Instrumente de măsurare a complexității unui algoritm. Complexitatea temporală. Complexitatea spațială. Ordine de complexitate. Notățiile asimptotice O , Θ și Ω . Exemple. 1.2. Demonstrarea modului de analiză a complexității algoritmilor pentru probleme specifice.	3	Expunerea, prelegerea, conversația, exemplificarea, demonstrația	
2. Metoda Divide et Impera: introducere 2.1. Prezentarea principiilor metodei Divide et Impera. 2.2. Introducerea metodei Divide et Impera prin discutarea și analiza unor probleme exemplu (<i>e.g.</i> , determinarea valorii extreme dintr-o mulțime de numere reale, algoritmul căutării binare) și realizarea de comparații cu alte soluții algoritmice pentru rezolvarea acestor probleme. 2.3. Analiza complexității algoritmilor Divide et Impera. Teorema Master. Exemple de aplicare a teoremei Master pentru diverse probleme.	3		
3. Metoda Divide et Impera: aplicații 3.1. Algoritmul sortării prin interclasare. 3.2. Algoritmul sortării rapide. 3.3. Analiza comparativă a performanțelor diverselor metode de sortare a datelor. Realizarea de conexiuni cu alți algoritmi studiați în cadrul unor discipline anterioare. Identificarea algoritmilor de sortare implementați în cadrul diverselor platforme de dezvoltare a aplicațiilor software. 3.4. Demonstrarea aplicării metodei Divide et Impera pentru alte probleme specifice, <i>e.g.</i> , problema determinării numărului de inversiuni ale unei permutări, problema tăieturilor, probleme de partiționare a figurilor geometrice, etc.	3		
4. Metoda programării dinamice: introducere 4.1. Prezentarea principiilor metodei programării dinamice. Rezolvarea problemelor algoritmice folosind principiile programării dinamice. 4.2. Exemplificarea metodei prin analiza unor probleme specifice, <i>e.g.</i> , șirul lui Fibonacci, calculul combinărilor $C_{n,k}$. Realizarea de comparații cu alte soluții pentru rezolvarea acestor probleme și analiza complexității diverselor abordări.	3		
5. Metoda programării dinamice: aplicații (1) 5.1. Problema subsecvenței de sumă maximă. Discutarea diverselor soluții algoritmice și analiza complexității acestora. 5.2. Rezolvarea eficientă a problemelor de natură algoritmică folosind structuri de tip Lookup Table (LUT) sau Summed Area Table (SAT).	3		

6. Metoda programării dinamice: aplicații (2) 6.1. Problema celei mai lungi subsecvențe crescătoare (LIS/LAS) și aplicații privind rezolvarea de probleme asemănătoare întâlnite în practică. 6.2. Problema celei mai lungi subsecvențe de elemente comune (LCS) și aplicații privind rezolvarea de probleme asemănătoare întâlnite în practică. Analiza complexității algoritmice a soluțiilor problemelor prezentate.	3	Expunerea, prelegerea, conversația, exemplificarea, demonstrația		
7. Metoda programării dinamice: aplicații (3) 7.1. Algoritmi eficienți pentru determinarea drumurilor optime în vederea parcurgerii unui labirint. 7.2. Rezolvarea problemelor de labirint și aplicații privind probleme asemănătoare întâlnite în practică. Analiza complexității algoritmice a soluțiilor problemelor prezentate.	3			
8. Algoritmi în grafuri: introducere 8.1. Modalități de reprezentare a grafurilor: matricea de adiacență și liste de adiacență. Exemple. 8.2. Tehnici de parcurgere a informației dintr-un graf: algoritmul parcurgerii în lățime (BFS) și algoritmul parcurgerii în adâncime (DFS).	3			
9. Algoritmi în grafuri: parcurgeri, componente conexe 9.1. Probleme practice rezolvate folosind arborele parcurgerii în adâncime. 9.2. Determinarea componentelor conexe ale unui graf neorientat. Analiza complexității algoritmice. Rezolvarea de probleme exemplu.	3			
10. Algoritmi în grafuri: drumuri 10.1. Drumuri în graf. Drumuri de cost minim. 10.2. Algoritmul Roy-Floyd-Warshall pentru determinarea tuturor drumurilor de cost minim din cadrul unui graf. 10.3. Algoritmul lui Dijkstra. 10.4. Aplicații ale determinării drumurilor într-un graf pentru rezolvarea de probleme specifice.	3			
11. Algoritmi în grafuri: cicluri 11.1. Cicluri în graf. Cicluri Hamiltoniene. Cicluri Euleriene. 11.2. Analiza performanțelor algoritmilor de determinare a ciclurilor unui graf.	3			
12. Algoritmi în grafuri: arbori parțiali 12.1. Arbori parțiali. Arbori parțiali de cost minim. 12.2. Algoritmul lui Prim. 12.3. Algoritmul lui Kruskal.	3			
13. Algoritmi geometrici 13.1. Introducere în geometria computațională. Reprezentarea datelor geometrice. 13.2. Problema intersecției segmentelor de dreaptă. 13.3. Problema intersecției figurilor geometrice. 13.4. Calculul ariilor figurilor geometrice. 13.5. Reprezentarea poligoanelor convexe și concave. 13.6. Ordonarea punctelor în plan. Algoritmul înfășurătorii convexe a unei mulțimi de puncte.	3			
14. Algoritmi pentru șiruri de caractere 14.1. Potrivirea șirurilor de caractere și algoritmul naiv de potrivire. 14.2. Algoritmul Rabin-Karp. 14.3. Algoritmul Boyer-Moore.	3			
Bibliografie				
1. Thomas H. Cormen, Charles E. Leiserson, Ronald R. Rivest. 2019. Introduction to Algorithms, 3rd Ed. MIT Press. (sau traducerea în limba română: Introducere în algoritmi, Ed. Agora)				

2. Donald E. Knuth. 2011. The Art of Computer Programming, vol. 1, 1st Ed., Addison-Wesley Professional. (Traducere în limba română: Arta programării calculatoarelor, vol. I – Algoritmi Fundamentali, Ed. Teora, 1999)
3. Donald E. Knuth. 2011. The Art of Computer Programming, vol. 2, 1st Ed., Addison-Wesley Professional. (Traducere în limba română: Arta programării calculatoarelor, vol. II – Algoritmi Seminumerici, Ed. Teora, 1999)
4. Donald E. Knuth. 2011. The Art of Computer Programming, vol. 3, 1st Ed., Addison-Wesley Professional. (Traducere în limba română: Arta programării calculatoarelor, vol. III – Sortare și căutare, Ed. Teora, 1999)
5. Matthias Felleisen, Robert Bruce Findler, Matthew Flatt, Shriram Krishnamurthi. 2018. How to Design Programs: An Introduction to Programming and Computing, 2nd Ed., MIT Press
6. Robert Sedgewick, Kevin Wayne. 2011. Algorithms, 4th Ed. Addison-Wesley Professional
7. Gayle Laakmann McDowell. 2015. Cracking the Coding Interview: 189 Programming Questions and Solutions, 6th Ed., CareerCup
8. Yonghui Wu, Jiande Wang. 2016. Data Structure Practice for Collegiate Programming Contests and Education, 1st Ed., CRC Press
9. Cristian A. Giumale. 2004. Introducere în analiza algoritmilor. Ed. Polirom
10. Cristian A. Giumale. 2000. Un atelier de programare. Ed. Agora
11. Radu-Daniel Vatavu. 2013. Ghid practic pentru analiza și proiectarea algoritmilor. Aplicații în C#. MatrixRom
12. Ștefan-Gheorghe Pentiuc, Radu-Daniel Vatavu. 2009. Algoritmi și metode de programare în Java. Ed. Universității Ștefan cel Mare Suceava
13. Doina Logofătu. 2007. Algoritmi fundamentali în C++: Aplicații. Ed. Polirom
14. Mihai Oltean. 1999. Proiectarea și implementarea algoritmilor. Ed. Agora
15. Radu-Daniel Vatavu, Lisa Anthony, Jacob O. Wobbrock. 2018. \$Q: A super-quick, articulation-invariant stroke-gesture recognizer for low-resource devices. In Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18). ACM, New York, NY, USA
16. Radu-Daniel Vatavu. 2017. Improving Gesture Recognition Accuracy on Touch Screens for Users with Low Vision. In Proceedings of the CHI Conference on Human Factors in Computing Systems. ACM, USA, 4667-4679
17. Yosra Rekik, Radu-Daniel Vatavu, Laurent Grisoni. 2014. Match-Up & Conquer: A Two-Step Technique for Recognizing Unconstrained Bimanual and Multi-Finger Touch Input. In Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14). ACM, New York, NY, USA, 201-208
18. Radu-Daniel Vatavu, Lisa Anthony, Jacob O. Wobbrock. 2012. Gestures as point clouds: A \$P recognizer for user interface prototypes. Proceedings of the ACM International Conference on Multimodal Interaction. ACM, 273-280

Bibliografie minimală

1. Thomas H. Cormen, Charles E. Leiserson, Ronald R. Rivest. 2019. Introduction to Algorithms, 3rd Ed. MIT Press. (sau traducerea în limba română: Introducere în algoritmi, Ed. Agora)
2. Gayle Laakmann McDowell. 2015. Cracking the Coding Interview: 189 Programming Questions and Solutions, 6th Ed., CareerCup
3. Radu-Daniel Vatavu. 2013. Ghid practic pentru analiza și proiectarea algoritmilor. Aplicații în C#, MatrixRom

Laborator	Nr. ore	Metode de predare	Observații
1. Analiza complexității algoritmilor (1)	2	Lucrări practice, experimentul, conversația, studiul de caz	
2. Analiza complexității algoritmilor (2)	2		
3. Metoda Divide et Impera: Prezentare generală	2		
4. Metoda Divide et Impera: Algoritmul sortării prin interclasare. Algoritmul sortării rapide	2		
5. Test practic și evaluare	2	Test practic	Rezolvarea unei probleme algoritmice prin implementarea corectă într-un limbaj de programare
6. Metoda programării dinamice: Introducere. Subsecvența crescătoare de lungime maximă	2	Lucrări practice, experimentul, conversația, studiul de caz	
7. Metoda programării dinamice: Subsecvența comună de lungime maximă	2		
8. Metoda programării dinamice: Parcurgerea labirintului	2		
9. Test practic și evaluare	2	Test practic	Rezolvarea unei probleme algoritmice prin implementarea corectă într-un limbaj de programare
10. Grafuri: Introducere	2	Lucrări practice, experimentul, conversația, studiul de caz	
11. Grafuri: Drumuri de cost minim	2		
12. Grafuri: Arbori parțiali de cost minim	2		

13. Test practic și evaluare	2	Test practic	Rezolvarea unei probleme algoritmice prin implementarea corectă într-un limbaj de programare
14. Rezolvarea de probleme algoritmice specifice	2	Lucrări practice, test practic, experimentul	

Prezența la activitățile obligatorii este reglementată de “Regulamentul cadru privind evaluarea studenților”. Recuperarea activităților aplicative obligatorii programate pe parcursul semestrului se face în conformitate cu precizările aceluiași regulament. De asemenea, în conformitate cu prevederile articolului 5.2.3, activitățile pe parcurs pot fi echivalate cu proiecte din tematica disciplinei cu acordul cadrului didactic titular.

Bibliografie

1. Thomas H. Cormen, Charles E. Leiserson, Ronald R. Rivest. 2019. Introduction to Algorithms, 3rd Ed. MIT Press. (sau traducerea în limba română: Introducere în algoritmi, Ed. Agora)
2. Donald E. Knuth. 2011. The Art of Computer Programming, vol. 1, 1st Ed., Addison-Wesley Professional. (Traducere în limba română: Arta programării calculatoarelor, vol. I – Algoritmi Fundamentali, Ed. Teora, 1999)
3. Donald E. Knuth. 2011. The Art of Computer Programming, vol. 2, 1st Ed., Addison-Wesley Professional. (Traducere în limba română: Arta programării calculatoarelor, vol. II – Algoritmi Seminumerici, Ed. Teora, 1999)
4. Donald E. Knuth. 2011. The Art of Computer Programming, vol. 3, 1st Ed., Addison-Wesley Professional. (Traducere în limba română: Arta programării calculatoarelor, vol. III – Sortare și căutare, Ed. Teora, 1999)
5. Matthias Felleisen, Robert Bruce Findler, Matthew Flatt, Shriram Krishnamurthi. 2018. How to Design Programs: An Introduction to Programming and Computing, 2nd Ed., MIT Press
6. Robert Sedgewick, Kevin Wayne. 2011. Algorithms, 4th Ed. Addison-Wesley Professional
7. Gayle Laakmann McDowell. 2015. Cracking the Coding Interview: 189 Programming Questions and Solutions, 6th Ed., CareerCup
8. Yonghui Wu, Jiande Wang. 2016. Data Structure Practice for Collegiate Programming Contests and Education, 1st Ed., CRC Press
9. Cristian A. Giumale. 2004. Introducere în analiza algoritmilor. Ed. Polirom
10. Cristian A. Giumale. 2000. Un atelier de programare. Ed. Agora
11. Radu-Daniel Vatavu. 2013. Ghid practic pentru analiza și proiectarea algoritmilor. Aplicații în C#. MatrixRom
12. Ștefan-Gheorghe Pentiuc, Radu-Daniel Vatavu. 2009. Algoritmi și metode de programare în Java. Ed. Universității Ștefan cel Mare Suceava
13. Doina Logofătu. 2007. Algoritmi fundamentali în C++: Aplicații. Ed. Polirom
14. Mihai Oltean. 1999. Proiectarea și implementarea algoritmilor. Ed. Agora
15. Radu-Daniel Vatavu, Lisa Anthony, Jacob O. Wobbrock. 2018. \$Q: A super-quick, articulation-invariant stroke-gesture recognizer for low-resource devices. In Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18). ACM, New York, NY, USA
16. Radu-Daniel Vatavu. 2017. Improving Gesture Recognition Accuracy on Touch Screens for Users with Low Vision. In Proceedings of the CHI Conference on Human Factors in Computing Systems. ACM, USA, 4667-4679
17. Yosra Rekik, Radu-Daniel Vatavu, Laurent Grisoni. 2014. Match-Up & Conquer: A Two-Step Technique for Recognizing Unconstrained Bimanual and Multi-Finger Touch Input. In Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces (AVI '14). ACM, New York, NY, USA, 201-208
18. Radu-Daniel Vatavu, Lisa Anthony, Jacob O. Wobbrock. 2012. Gestures as point clouds: A \$P recognizer for user interface prototypes. Proceedings of the ACM International Conference on Multimodal Interaction. ACM, 273-280

Bibliografie minimală

1. Thomas H. Cormen, Charles E. Leiserson, Ronald R. Rivest. 2019. Introduction to Algorithms, 3rd Ed. MIT Press. (sau traducerea în limba română: Introducere în algoritmi, Ed. Agora)
2. Gayle Laakmann McDowell. 2015. Cracking the Coding Interview: 189 Programming Questions and Solutions, 6th Ed., CareerCup
3. Radu-Daniel Vatavu. 2013. Ghid practic pentru analiza și proiectarea algoritmilor. Aplicații în C#, MatrixRom

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

Conținutul disciplinei este în concordanță cu conținutul disciplinelor similare de la programele de studiu din cadrul altor universități din țară și străinătate, și anume “Introduction to Algorithms”, MIT, MA, USA (<https://ocw.mit.edu/courses/6-006-introduction-to-algorithms-spring-2020>); “Intro to Algorithms and their Limitations”, Harvard University, MA, USA (<https://salil.seas.harvard.edu/classes/intro-algorithms-and-their-limitations-fall21>); „Introduction to Algorithms”, Washington University, WA, USA (<https://courses.cs.washington.edu/courses/cse421>); „Proiectarea algoritmilor”, Universitatea Politehnica București, (<https://ocw.cs.pub.ro/courses/pa>). De asemenea, conținutul cursului este în acord cu recomandările ACM/IEEE privind Computer Engineering Curricula- Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering în ce privește “1. Fundamental algorithmic design principles, 2. Analysis of algorithmic behavior,

including tradeoffs between algorithms, 3. Classic algorithms for such common tasks as searching and sorting, 4. Design and analysis of application-specific algorithms” (p. 71),
<https://www.acm.org/binaries/content/assets/education/ce2016-final-report.pdf>.

10. Evaluare

Tip activitate	Criterii de evaluare	Metode de evaluare	Pondere din nota finală
Curs	Expunerea clară a conceptelor teoretice și corectitudinea operării cu noțiuni specifice proiectării algoritmilor ca fundamente științifice, ingineresti și ale informaticii. Rezolvarea corectă a problemelor de natură algoritmică pentru prelucrarea datelor în diverse reprezentări și pentru diverse aplicații. Analiza corectă a complexității teoretice a algoritmilor în vederea îmbunătățirii performanțelor sistemelor software.	Examen scris. Întrebări adresate în timpul orelor de curs.	50%
Seminar	-	-	-
Laborator	Corectitudinea rezolvării de probleme de natură algoritmică și a implementării acestora folosind un limbaj de programare și o platformă de dezvoltare a aplicațiilor software. Gradul de îndeplinire a activităților din lucrările practice de laborator și realizarea de teme suplimentare. Participarea la diverse activități practice în legătură cu obiectivele specifice ale disciplinei.	Teste practice pe calculator. Întrebări adresate în timpul orelor de laborator.	50%
Proiect	-	-	-
Standard minim de performanță			
<ul style="list-style-type: none"> - Însușirea principalelor noțiuni, idei, concepte din cadrul proiectării și analizei algoritmilor - Cunoașterea problemelor de bază din domeniul proiectării și analizei algoritmilor - Implementarea de soluții algoritmice pentru problemele de bază din domeniul proiectării algoritmilor 			

Data completării	Semnătura titularului de curs	Semnătura titularului de aplicație
22.09.2022		

Data avizării în departament	Semnătura directorului de departament
29.09.2022	

Data aprobării în consiliul facultății	Semnătura decanului
30.09.2022	