

FIȘA DISCIPLINEI

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea „Tibiscus” din Timișoara
1.2. Facultatea	Calculatoare și Informatică Aplicată
1.3. Departamentul	Informatică
1.4. Domeniul de studii	Informatică
1.5. Ciclul de studii	Licență
1.6. Programul de studii/Calificarea	Informatică

2. Date despre disciplină

2.1. Denumirea disciplinei	PROGRAMARE LOGICĂ ȘI FUNCȚIONALĂ (PLF) – LIN323						
2.2. Titularul activității de curs	Conf.univ.dr. Florentina-Anica PINTEA						
2.3. Titularul activității de seminar	Inf. Dan PĂTRAȘCU-BABA						
2.4. Anul de studiu	3	2.5. Semestrul	2	2.6. Tipul de evaluare	E	2.7. Regimul disciplinei	Conținut ¹ =DS Obligatorietate ² =DO

3. Timpul total estimat

3.1. Numărul de ore pe săptămână	4	din care 3.2. curs	2	3.3. laborator	2
3.4. Total ore din planul de învățământ	48	din care 3.5. curs	24	3.6. laborator	24
Distribuția fondului de timp					Ore
Studiul după manual, suport de curs, bibliografie și notițe					12
Documentare suplimentară în bibliotecă, pe platforme electronice de specialitate și pe teren					10
Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri					14
Tutoriat					12
Examinări					4
Alte activități					-
3.7. Total ore studiu individual	52				
3.8. Total ore pe semestru	100				
3.9. Numărul de credite	4				

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	Cunoștințe fundamentale de programare (structuri de control, funcții, recursivitate, tipuri de date). Noțiuni de structuri de date și algoritmi (liste, arbori, căutare, parcurgere). Elemente de matematică discretă (logică propozițională și predicativă, relații, funcții, inducție matematică).
4.2. de competențe	Capacitatea de a analiza și formaliza o problemă în termeni algoritmici. Abilitatea de a utiliza un limbaj de programare procedural sau orientat pe obiect. Gândire logică și capacitate de abstractizare.

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sală de curs dotată cu videoproiector. Calculator/laptop pentru prezentarea suportului de curs și a demonstrațiilor. Acces la internet pentru consultarea documentației și a resurselor online. Suport de curs disponibil în format electronic pe platforma e-learning a facultății.
5.2. de desfășurare a seminarului/laboratorului	Laborator informatic dotat cu calculatoare funcționale pentru fiecare student sau lucru în echipe de maximum 2 studenți.

	Instalarea mediilor de dezvoltare specifice limbajelor utilizate (ex.: SWI-Prolog sau alt limbaj funcțional ales). Acces la internet pentru documentație și exemple de cod. Platformă de încărcare și evaluare a temelor (Google Classroom).
--	---

6. Obiectivele disciplinei - rezultate așteptate ale învățării la formarea cărora contribuie parcurgerea și promovarea disciplinei

Cunoștințe	La finalizarea disciplinei, studentul va cunoaște și va înțelege: – Fundamentele teoretice ale paradigmelor de programare logică și funcțională. – Diferențele conceptuale dintre paradigma imperativă, funcțională și logică. – Concepte specifice programării logice: fapte, reguli, interogări, unificare, backtracking. – Mecanisme de recursivitate și utilizarea tipurilor algebrice de date. – Modele de rezolvare declarativă a problemelor.
Abilități	La finalizarea disciplinei, studentul va fi capabil să: - Rezolve probleme utilizând paradigma funcțională și logică. - Scrie programe corecte și eficiente în limbaje precum Prolog, Lisp - Analizeze și depaneze programe declarative.
Responsabilitate și autonomie	Capacitatea de a lucra autonom în proiectarea și implementarea soluțiilor declarative. Capacitatea de a documenta și argumenta soluțiile propuse. Responsabilitate în respectarea standardelor de cod și a termenelor de predare.

7. Conținuturi

7.1. Curs	Metode de predare	Observații
1. Noțiuni de bază în programarea funcțională (structuri de date, liste, mapare, recursivitate) 2. Recursivitate în Lisp. Acumulatori 3. Backtracking și predicatul de tăiere. Negația 4. Backtracking și predicatul de tăiere 5. I/O în Prolog. Lucrul cu fișiere. Definirea de noi operatori 6. Logica predicatelor de ordinul I. Forma clauzala. Unificare. Fapte. Reguli. sintaxă 7. Introducere în Clisp, variabile, atribuire, recursivitate 8. Funcții final recursive 9. Iterativitate 10. Liste circulare, Liste de asociere 11. Închideri 12. Probleme de scriere macro-uri	Expunerea interactivă, problematizarea, conversația euristică, documentarea pe web, exemplificarea	Materialele de curs prezentate la curs vor disponibile și pe platforma pe Google Classroom
Bibliografie 1. Greg Michaelson, An Introduction to Functional Programming Through Lambda Calculus - (fundamente teoretice ale programării funcționale) / https://www.goodreads.com/shelf/show/functional-programming 2. M. Oprea (2021), Programare logică și programare funcțională. Teorie și aplicații - manual adaptat pe concepte declarative și logice. 3. M. Oprea (1999), Inteligență artificială – Programare logică și Sisteme expert, vol. II, UPG Ploiești. 4. R. Bird (1998), Introduction to Functional Programming, 2nd edition, Prentice Hall, New York. 5. C. Giumale (1997), Programare funcțională - o perspectivă pragmatică, Editura Tehnică, București.		
7.2. Seminar/laborator	Metode de seminarizare	Observații
1. Sintaxa și structuri de date, liste în Prolog 2. Recursivitate 3. Acumulatori, backtracking, predicatul cut 4. I/O Operații cu fișiere. Definire de noi operatori 5. Limbaj natural. Eficiența în Prolog. Stringuri 6. Proiect în echipă - recapitulare Prolog 7. Introducere în Clisp 8. Definire de funcții, IF, COND 9. Recursivitate, structuri de control 10. Expresii lambda, argumente, liste 11. Mini-interpreter Lisp, mini-server TCP 12. Macro-uri, utilizare, exemple	Exercițiul, discuțiile și dezbaterile, modelarea, proiectul, lucrul în grup organizat	Temele pe care studenții le vor avea de rezolvat după exemplele de laborator, vor fi disponibile pe Google Classroom, iar fiecare student va trebui să încarce temele rezolvate

Bibliografie

1. **How to Design Programs (2nd Edition)** – Felleisen et al. — abordare modernă a designului programelor funcționale. / https://en.wikipedia.org/wiki/How_to_Design_Programs
2. <https://common-lisp.net/tutorials/> - Common-Lisp
3. <http://www.tutorialspoint.com/lisp/> - LISP Tutorial
4. https://www.cpp.edu/~jrfisher/www/prolog_tutorial/contents.html - Prolog tutorial
5. *Structure and Interpretation of Computer Programs* / https://en.wikipedia.org/wiki/Structure_and_Interpretation_of_Computer_Programs

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

Conținutul disciplinei este corelat cu standardele academice și profesionale din domeniul Informaticii, fiind aliniat la recomandările comunității epistemice internaționale privind formarea competențelor fundamentale în paradigme declarative și funcționale de programare.

Cursul dezvoltă deprinderi de modelare, care sunt utile dincolo de paradigma de programare logică și cea de programare funcțională, deprinderi care vor fi utile în activitatea profesională pe piața muncii.

Programarea funcțională este utilizată în tehnologii moderne precum și concepte funcționale integrate în Java, Python sau JavaScript), iar principiile programării logice stau la baza sistemelor de inteligență artificială, sistemelor expert și mecanismelor de inferență.

9. Evaluare

Tip de activitate	Criterii de evaluare	Metode de evaluare	Pondere din nota finală
9.1. Curs	Cunoașterea și înțelegerea conceptelor fundamentale ale programării logice și funcționale; capacitatea de analiză și rezolvare a problemelor	Examen scris (teorie + aplicații)	40%
9.2. Laborator	Aplicarea practică a conceptelor studiate; corectitudinea și eficiența programelor realizate; capacitatea de argumentare a soluțiilor	Proiect practic + prezentareParticiparea activă la activitățile de laborator	30%
	Temele sau proiectele acoperă părți ale materiei prezentate la laborator, în condiții similare examinării de laborator.	Proiect individual, dosar probleme	30%
9.3. Standard minim de performanță			
Examinare scrisă: • Cunoașterea principiilor de bază ale programării logice și funcționale - Pentru nota 5 este necesară capacitatea de a implementa corect algoritmi simpli în Prolog și în Lisp. • Capacitatea de a scrie și rula programe simple într-un limbaj funcțional și/sau logic • Realizarea unui proiect practic minimal funcțional care să rezolve o problemă utilizând conceptele studiate. • Studenții trebuie să aibe minim 50 % prezență la laborator.			

Notă:

1) Regimul disciplinei (conținut) - pentru nivelul de licență se alege una din variantele: DF (disciplină fundamentală) / DS (disciplină de specialitate) / DC (disciplină complementară).

2) Regimul disciplinei (obligativitate) - se alege una din variantele: DO=discipline obligatorii (impuse) / DOP=discipline opționale/ DFA (disciplină facultativă).

Data completării

12.02.2026

Titular de curs și laborator

Conf.univ.dr. Florentina-Anica PINTEA

Data avizării în departament

20.02.2026

Director de departament

Conf. univ. dr. Victoria IORDAN