FIȘA DISCIPLINEI

1. Date despre program

Facultatea	Inginerie Electrică și Știința Calculatoarelor
Departamentul	Calculatoare, electronică şi automatică
Domeniul de studii	Automatică, informatică aplicată şi sisteme inteligente
Ciclul de studii	Licență, dual
Programul de studii	Automatică şi informatică aplicată

2. Date despre disciplină

Denumirea disciplinei	**PROGRAMARE ORIENTATĂ PE OBIECTE**				
Anul de studiu	**II**	Semestrul	**3**	Tipul de evaluare	**E**
Regimul disciplinei	Categoria formativă a disciplinei DF - fundamentală, DS - de specializare, DC – complementară				**DF**
	Categoria de opționalitate a disciplinei: DOB – obligatorie, DOP – opțională, DFA - facultativă				**DOB**

3. Timpul total estimat (ore alocate activităților didactice)

I a) Număr de ore pe săptămână	Total general	4	Curs	**2**	Seminar		Laborator/ Lucrări practice IIS	**2**	Proiect IIS		Practică IIS	
							Laborator/ Lucrări practice IM		Proiect IM		Practică IM	
I b) Totalul de ore pe semestru din planul de învățământ		56	Curs	**28**	Seminar		Laborator	**28**	Proiect		Practică	

(*IIS – instituție de învățământ superior; IM – învățare prin muncă*)

Distribuția fondului de timp pe semestru	Ore IIS	Ore IM
II.Studiu individual	**16**	
III. Examinări	**3**	
IV. Alte activități (precizați):		

Total ore studiu individual (II + III)	Ore IIS	**19**	Ore IM	
Total ore pe semestru (Ib+II+III+IV)	Ore IIS	**75**	Ore IM	
Numărul de credite	Credite IIS	**3**	Credite IM	

4. Competențe specifice acumulate

Competențe transversale	CT6. Își asumă responsabilitatea
Competențe profesionale	CP9. Dezvoltă aplicații de procesare de date CP15. Utilizează programare pe bază de scripturi CP16. Creează softuri

5. Rezultatele învățării

Cunoștințe	Aptitudini	Responsabilitate și autonomie
Studentul/absolventul explică și interpretează conceptele programării orientate pe obiecte (abstractizare, încapsulare, moștenire, polimorfism), mecanismele limbajului C++ (șabloane, excepții, gestiunea resurselor) și comportamentul programelor C++ la compilare și execuție.	Studentul/absolventul concepe soluții orientate pe obiecte pentru probleme de inginerie de complexitate medie, le implementează în C++ folosind ierarhii de clase, șabloane și STL și le validează prin teste unitare. Studentul/absolventul aplică tehnici de management de proiect în dezvoltarea în echipă a unei	Studentul/absolventul lucrează eficient ca membru în echipă sau lider al acesteia în proiectul de laborator, cu contribuții transparente în istoricul Git. Studentul/absolventul aplică valorile eticii și deontologiei profesiei de inginer, respectând licențele software și declarând utilizarea instrumentelor de inteligență artificială.

	aplicații C++, utilizând un sistem de construire (CMake) și controlul versiunilor (Git).	

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

Obiectivul general al disciplinei	Însușirea paradigmei programării orientate pe obiecte și a limbajului C++ modern, inclusiv biblioteca standard STL, și formarea deprinderilor de dezvoltare, versionare și testare a proiectelor software de complexitate medie.

7. Conținutul predării și învățării

Curs	Nr. ore	Metode de predare	Observații
1. Introducere în programarea orientată pe obiecte și în limbajul C++ 1.1. Paradigme de programare: procedurală vs. orientată pe obiecte. Abstractizare, încapsulare, moștenire, polimorfism 1.2. Limbajul C++: evoluție și standarde (C++11 – C++23), compilare și medii de dezvoltare 1.3. Extensii față de C: referințe, spații de nume, const/constexpr, auto, supraîncărcarea funcțiilor, parametri impliciți 1.4. Alocarea dinamică cu new/delete. Fluxurile standard cin/cout	2	expunerea, prelegerea-dezbatere, demonstrația	
2. Clase și obiecte 2.1. Declararea claselor. Date și funcții membre. Specificatori de acces 2.2. Constructori (implicit, cu parametri, de copiere, de mutare), liste de inițializare. Destructori 2.3. Pointerul this. Membri statici și membri constanți 2.4. Funcții și clase prietene 2.5. Gestiunea resurselor prin obiecte (RAII)	4	expunerea, prelegerea-dezbatere, demonstrația	
3. Supraîncărcarea operatorilor 3.1. Reguli de supraîncărcare. Operatori ca funcții membre sau prietene 3.2. Operatori aritmetici, relaționali, de indexare, de apel și de inserție/extracție în flux 3.3. Semantica copierii și a mutării. Regula celor 3/5/0	2	expunerea, prelegerea-dezbatere, demonstrația	
4. Moștenirea 4.1. Clase de bază și clase derivate. Specificatori de acces la derivare 4.2. Constructori și destructori în ierarhii de clase 4.3. Redefinirea membrilor. Conversii între clasa de bază și clasa derivată 4.4. Moștenirea multiplă și moștenirea virtuală 4.5. Compunere vs. moștenire	4	expunerea, prelegerea-dezbatere, demonstrația	
5. Polimorfismul 5.1. Legarea statică și legarea dinamică. Funcții virtuale și tabela de funcții virtuale 5.2. Clase abstracte și funcții virtuale pure. Interfețe 5.3. Destructori virtuali. Specificatorii override și final. RTTI și dynamic_cast	2	expunerea, prelegerea-dezbatere, demonstrația	
6. Programarea generică: șabloane 6.1. Șabloane de funcții și șabloane de clase 6.2. Specializarea șabloanelor 6.3. Constrângeri asupra parametrilor șablon: concepte (C++20)	2	expunerea, prelegerea-dezbatere, demonstrația	
7. Excepții și gestiunea resurselor 7.1. Tratarea excepțiilor: try, catch, throw. Ierarhia std::exception 7.2. Garanții de siguranță la excepții 7.3. Pointeri inteligenți: unique_ptr, shared_ptr, weak_ptr	2	expunerea, prelegerea-dezbatere, demonstrația	
8. Fluxuri de intrare/ieșire 8.1. Ierarhia claselor de fluxuri. Formatarea intrărilor/ieșirilor 8.2. Fluxuri pentru fișiere text și binare. Fluxuri pe șiruri (stringstream) 8.3. Serializarea obiectelor	2	expunerea, prelegerea-dezbatere, demonstrația	
9. Biblioteca standard de șabloane (STL) 9.1. Containere secvențiale: vector, array, deque, list 9.2. Containere asociative: set, map, unordered_set, unordered_map 9.3. Iteratori și adaptori de containere 9.4. Algoritmi standard: căutare, sortare, transformare, acumulare 9.5. Funcții lambda și obiecte funcție 9.6. Clase utilitare: string, string_view, optional. Biblioteca ranges (C++20)	4	expunerea, prelegerea-dezbatere, demonstrația	
10. Dezvoltarea proiectelor software complexe 10.1. Organizarea proiectelor: fișiere antet, spații de nume, module. Construirea cu CMake 10.2. Controlul versiunilor cu Git: commit, ramuri, îmbinare, rezolvarea conflictelor, pull request (GitHub/GitLab) 10.3. Testarea: teste unitare (GoogleTest/Catch2), dezvoltare ghidată de teste (TDD), depanare, analiză statică și sanitizers 10.4. Principii de proiectare (SOLID) și șabloane de proiectare (Factory, Singleton, Observer, Strategy). Modelare UML 10.5. Documentarea codului (Doxygen) și integrarea continuă (CI)	4	expunerea, prelegerea-dezbatere, demonstrația	

Bibliografie minimală recomandată
1. Bjarne Stroustrup, A Tour of C++ (3rd Edition), Addison-Wesley, ISBN-13: 978-0136816485, 2022; 2. Bjarne Stroustrup, Programming: Principles and Practice Using C++ (3rd Edition), Addison-Wesley, ISBN-13: 978-0138308681, 2024; 3. Bjarne Stroustrup, The C++ Programming Language (4th Edition), Addison-Wesley, ISBN-13: 978-0321563842, 2013; 4. Nicolai M. Josuttis, The C++ Standard Library: A Tutorial and Reference (2nd Edition), Addison-Wesley, ISBN-13: 978-0321623218, 2012; 5. Scott Meyers, Effective Modern C++, O'Reilly Media, ISBN-13: 978-1491903995, 2014; 6. Björn Andrist, Viktor Sehr, C++ High Performance (2nd Edition), Packt Publishing, ISBN-13: 978-1839216541, 2020; 7. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, ISBN-13: 978-0201633610, 1994; 8. Scott Chacon, Ben Straub, Pro Git (2nd Edition), Apress, 2014, https://git-scm.com/book; 9. Bjarne Stroustrup, C++, Editura Teora, ISBN: 973-20-0894-6, 2003; 10. Ioan Salomie, Tehnici orientate pe obiecte, Editura Microinformatica, Cluj-Napoca, 1995; 11. Remus Prodan – pagina web a cursului: http://dev.fiesc.usv.ro/remus; 12. Referința limbajului C++: https://en.cppreference.com.

Aplicații (seminar / laborator / lucrări practice / proiect)	Nr. ore	Metode de predare	Observații
1. Prezentarea generală a laboratorului în care se vor desfășura activitățile practice pentru disciplina Programare orientată pe obiecte. Prezentarea normelor de protecția muncii. Familiarizarea cu mediul de lucru: compilator C++, IDE, CMake, Git	2	expunere, verificare/testare	
2. Elemente C++ neorientate pe obiecte: referințe, spații de nume, supraîncărcarea funcțiilor, new/delete	2	lucrări practice, experimentul	
3. Clase și obiecte. Constructori și destructori	2	lucrări practice, experimentul	
4. Constructorul de copiere și de mutare. Membri statici și constanți. Funcții prietene	2	lucrări practice, experimentul	
5. Supraîncărcarea operatorilor	2	lucrări practice, experimentul	
6. Moștenirea simplă și multiplă	2	lucrări practice, experimentul	
7. Funcții virtuale, clase abstracte, polimorfism. Testare programare în C++	2	lucrări practice, experimentul, verificare/testare	
8. Șabloane de funcții și de clase	2	lucrări practice, experimentul	
9. Tratarea excepțiilor. Pointeri inteligenți	2	lucrări practice, experimentul	
10. Fluxuri și fișiere. Serializarea obiectelor	2	lucrări practice, experimentul	
11. STL: containere și iteratori	2	lucrări practice, experimentul	
12. STL: algoritmi și funcții lambda. Testare programare în C++	2	lucrări practice, experimentul, verificare/testare	
13. Proiect: organizarea cu CMake, versionarea cu Git, teste unitare	2	lucrări practice, experimentul	
14. Proiect: prezentarea și susținerea proiectelor	2	lucrări practice, experimentul, verificare/testare	

Proiectul se realizează pe parcursul laboratorului, individual sau în echipe de 2-3 studenți, și constă în dezvoltarea unei aplicații C++ orientate pe obiecte de complexitate medie, împreună cu documentația aferentă. Etape: analiza cerințelor și modelarea claselor (UML); organizarea proiectului (CMake) și a depozitului Git; implementarea incrementală, cu contribuții vizibile în istoricul Git; testarea cu teste unitare și depanarea; documentarea și prezentarea proiectului.
Exemple de teme: gestiunea unei biblioteci, a unui magazin sau a situației școlare; jocuri în mod consolă (X și 0, șah, labirint); editor de figuri geometrice cu ierarhii de clase; simulator de circuite logice; aplicație pentru prelucrarea unor fișiere de date (CSV/JSON) folosind STL.

Bibliografie minimală recomandată
1. Bjarne Stroustrup, A Tour of C++ (3rd Edition), Addison-Wesley, ISBN-13: 978-0136816485, 2022; 2. Bjarne Stroustrup, Programming: Principles and Practice Using C++ (3rd Edition), Addison-Wesley, ISBN-13: 978-0138308681, 2024; 3. Bjarne Stroustrup, The C++ Programming Language (4th Edition), Addison-Wesley, ISBN-13: 978-0321563842, 2013; 4. Nicolai M. Josuttis, The C++ Standard Library: A Tutorial and Reference (2nd Edition), Addison-Wesley, ISBN-13: 978-0321623218, 2012; 5. Scott Meyers, Effective Modern C++, O'Reilly Media, ISBN-13: 978-1491903995, 2014; 6. Björn Andrist, Viktor Sehr, C++ High Performance (2nd Edition), Packt Publishing, ISBN-13: 978-1839216541, 2020; 7. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, ISBN-13: 978-0201633610, 1994; 8. Scott Chacon, Ben Straub, Pro Git (2nd Edition), Apress, 2014, https://git-scm.com/book; 9. Bjarne Stroustrup, C++, Editura Teora, ISBN: 973-20-0894-6, 2003; 10. Ioan Salomie, Tehnici orientate pe obiecte, Editura Microinformatica, Cluj-Napoca, 1995; 11. Remus Prodan – pagina web a cursului: http://dev.fiesc.usv.ro/remus; 12. Referința limbajului C++: https://en.cppreference.com.

Observații: - La cererea studentului, activitatea pe parcurs poate fi echivalată prin proiecte, pregătirea și participarea la concursuri profesionale sau alte tipuri de activități specifice, cu aprobarea cadrului didactic care conduce lucrările practice; - Studenții care au mai mult de 50% absențe la activitatea practică pot recupera această activitate cu aprobarea cadrului didactic care conduce lucrările practice.

8. Evaluare

Tip activitate	Criterii de evaluare	Metode de evaluare	Pondere din nota finală
Evaluare finală	Cunoștințe: corectitudinea și completitudinea explicării conceptelor POO, a mecanismelor C++ și a componentelor STL. Aptitudini: proiectarea și implementarea pe calculator a unei ierarhii de clase care rezolvă o cerință dată, folosind STL. Standard minim: nota 5 la fiecare probă; programul se compilează și rezolvă cerința de bază folosind clase proprii.	Examen: probă teoretică (scris sau oral) și probă practică pe calculator (elaborarea, testarea și explicarea unui program C++)	50
Evaluare pe parcurs	Aptitudini: corectitudinea programelor de laborator și a celor două testări pe calculator; calitatea proiectului (funcționalitate, arhitectură, teste). Responsabilitate și autonomie: susținerea proiectului, contribuția individuală vizibilă în Git, originalitatea codului și utilizarea declarată a instrumentelor de inteligență artificială. Standard minim: nota 5 la activitatea pe parcurs, cu proiectul predat și susținut.	Evaluare continuă la laborator; două testări practice pe calculator; susținerea proiectului	50

Fișa disciplinei include, dacă este cazul, elemente adaptate persoanelor cu dizabilități, în funcție de tipul și gradul acestora.
.

Data completării	Grad didactic, nume, prenume, semnătura titularului de curs	Grad didactic, nume, prenume, semnătura titularului de aplicație
18.09.2026	Ș.l. dr. ing. Remus Cătălin PRODAN	Ș.l. dr. ing. Remus Cătălin PRODAN Ș.l. dr. inf. Laura BILIUS

Data avizării	Grad didactic, nume, prenume, semnătura responsabilului de program
22.09.2026	Ș.l dr. ing. Corneliu BUZDUGA

Data avizării în departament	Grad didactic, nume, prenume, semnătura directorului de departament
24.09.2026	Conf. dr. ing. Eugen COCA

Data aprobării în consiliul facultății	Grad didactic, nume, prenume, semnătura decanului
25.09.2026	Prof. dr. ing. Laurentiu- Dan MILICI