

FIȘA DISCIPLINEI

Programare orientată obiect

Anul universitar 2026-2027

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea Babeș-Bolyai Cluj-Napoca
1.2. Facultatea	Facultatea de Matematică și Informatică
1.3. Departamentul	Departamentul de informatică
1.4. Domeniul de studii	Informatică
1.5. Ciclu de studii	Licență
1.6. Programul de studii / Calificarea	Inteligență artificială
1.7. Forma de învățământ	Cu frecvență

2. Date despre disciplină

2.1. Denumirea disciplinei	Programare orientată obiect			Codul disciplinei	MLE5006
2.2. Titularul activităților de curs	Lect. Dr. Ing. Diana-Laura Borza				
2.3. Titularul activităților de seminar	Lect. Dr. Ing. Diana-Laura Borza				
2.4. Anul de studiu	1	2.5. Semestrul	2	2.6. Tipul de evaluare	Examen
2.7. Regimul disciplinei	Obligatoriu		2.8. Tipul disciplinei	Disciplină fundamentală (DF)	

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	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)					15
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					15
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					15
Tutoriat (consiliere profesională)					5
Examinări					5
Alte activități					
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	Fundamentele Programării
4.2. de competențe	Capacitatea de programare în limbaje de nivel înalt

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sală de curs cu proiector
5.2. de desfășurare a seminarului/ laboratorului	Laborator cu calculatoare; medii de programare cu un compilator C++, librăria Qt instalată, doxygen, git

6.1. Competențele dobândite în urma absolvirii programului de studii (se preiau din planul de învățământ)¹

¹ Se vor prelua din Planul de învățământ al programului de studii acele competențe profesionale și/sau transversale la dezvoltarea cărora contribuie disciplina pentru care se elaborează fișa disciplinei. Pentru fiecare competență se va prelua întregul enunț, inclusiv codul competenței, cu formularea care apare în planul de

Competențe profesionale	
Codul competenței	Competență
CP1	Creează softuri
CP5	Definește cerințe tehnice
CP4	Definește arhitectura software
Competențe transversale	
Codul competenței	Competență
CT2	Soluționează probleme
CT3	Gândește analitic

6.2. Rezultatele învățării specifice programului de studii (se preiau din planul de învățământ)²

Rezultatele învățării vizate prin disciplină		
Codul competenței	Cunoștințe și înțelegere (Knowledge and understanding)	Abilități academice specifice (Specific academic skills)
CP2 CP3	Studentul/absolventul numește, oferă exemple, concluzionează, specifică, recunoaște și argumentează critic metodele de proiectare și management al proiectelor informatice complexe, utilizând strategii moderne	Studentul/absolventul inițiază, pregătește, realizează, propune metode de dezvoltare a proiectelor informatice complexe. Studentul/absolventul realizează rapoarte profesionale specifice
CP4	Studentul/absolventul cunoaște și explică paradigmele moderne de programare, arhitecturi software și metodologii de dezvoltarea proiectelor software.	Studentul/absolventul proiectează, planifică, construiește, dezvoltă aplicații software scalabile și utilizează eficient resursele hardware și software
CP11 CP12	Studentul/absolventul cunoaște șabloane de proiectare de software și metodologii de proiectare dirijată de utilizator pentru platforme desktop, mobile și web	Studentul/absolventul proiectează și dezvoltă aplicații software scalabile folosind practici moderne și șabloane de proiectare larg utilizate în industrie
CT1 CT2 CT3 CT4 CT5	Studentul/absolventul are cunoștințele necesare pentru a înțelege și soluționa probleme complexe, pentru a planifica și organiza procese avansate în diverse domenii	Absolventul este capabil să identifice probleme complexe și să examineze probleme conexe pentru a dezvolta opțiuni de rezolvare și implementa soluții. Absolventul are abilitatea de a aplica reguli generale unor probleme specifice și de a produce soluții relevante. Absolventul este capabil să combine informații diverse pentru a formula soluții și genera idei de dezvoltare pentru noi produse și aplicații.

7. Rezultatele învățării specifice disciplinei

Cunoștințe și înțelegere (Knowledge and understanding)
1. Să înțeleagă conceptele paradigmei de programare orientată pe obiecte și să proiecteze soluții orientate pe obiecte pentru probleme de mică/medie amploare, folosind C++ și Qt.
2. Să înțeleagă diferențele dintre proiectarea imperativă tradițională și proiectarea orientată pe obiecte.
3. Să explice clasele ca elemente fundamentale și modulare pentru dezvoltarea de programe.

învățământ, fără modificări. Dacă nu se preia nici o competență din oricare din cele două categorii, se șterge linia din tabel aferentă acelei categorii.

² Se menționează rezultatele învățării specifice programului de studiu la dezvoltarea cărora contribuie disciplina pentru care se elaborează fișa. Enunțurile, preluate fără modificări din Planul de învățământ în funcție de tipul disciplinei (DF/DS/DC) se trec în dreptul competenței asociate.

4. Să înțeleagă rolul moștenirii, polimorfismului, legării dinamice și al structurilor generice în construirea de cod reutilizabil.
Abilități academice specifice (Specific academic skills)
1. Să explice și să utilizeze strategii de programare defensivă, folosind aserțiuni formale și tratarea excepțiilor.
2. Să proiecteze interfețe cu utilizatorul și să scrie programe C++ de mică/medie amploare folosind Qt.
3. Să utilizeze clase scrise de alți programatori și biblioteci terțe în construirea sistemelor proprii.

8. Conținuturi

8.1 Curs	Metode de predare - învățare	Observații ³
Introducere în C++ (elementele de bază ale limbajului de programare C/C++, tipuri de date, variabile, constante, durata de viață a variabilelor, instrucțiuni, funcții: declarație și definiție, funcții de supraîncărcare).	Expunerea interactivă Explicația Conversația Demonstrația didactică	
Programare modulară în C/C++ (funcții, parametri formali și actuali, pointeri și managementul memoriei, stiva și <i>heap</i> , pointeri către funcții, fișiere <i>header</i> , programare modulară, biblioteci).		
Programare orientată obiect în C++ (introducere în programarea orientată pe obiecte, caracteristicile programării orientată obiect, abstractizare, încapsulare, clase și obiecte, modificatori de acces, crearea și distrugerea obiectelor, supraîncărcarea operatorilor, elemente statice și <i>friend</i>).		
Moștenire și polimorfism (clase de bază și derivate, principiul substituției Liskov, suprascrierea metodei, moștenirea și polimorfismul).		
Polimorfism (legare statică și dinamică, metode virtuale, moștenire multiplă, upcasting și downcasting, clase abstracte, diagrame de clasă UML și relații între clase).		
Programare generică în C++. Librăria standard C++ (C++ Standard Template Library) (șabloane de funcție (<i>function templates</i>), șabloane de clasă (<i>class templates</i>), containere în STL: tablou static, vector, listă, stivă, <i>heap</i> , <i>map</i> , mulțime), iteratoare, algoritmi STL, funcții lambda).		
Streams-uri și tratarea excepțiilor (fluxuri de ieșire de intrare, operatori de inserare și extracție, operatori de inserare și extracție de supraîncărcare, formatare, manipuloare, fișiere text, gestionarea excepțiilor, <i>exception safe code</i>).		

³ De exemplu aspecte organizatorice, recomandări pentru studenți, aspecte specifice legate de curs/seminar cum ar fi invitarea unor practicieni în domeniu etc.

Managementul resurselor și RAI (Resource Acquisition Is Initialization (RAI), <i>smart pointers, move semantics, std::unique_ptr, std::shared_ptr, std::weak_ptr</i>)		
Interfețe grafice în Qt (Qt Toolkit – descriere generală, component grafice Qt, Qt Designer).		
Programare bazată pe evenimente I (callback-uri, evenimente, semnale și sloturi în Qt).		
Programare bazată pe evenimente II (Model View Controller, Models și Views în Qt).		
Șabloane de proiectare I (șabloane de proiectare creaționale, structurale, comportamentale ; exemple; singleton, factory method, adapter).		
Șabloane de proiectare II (façade, observer, strategy, studii de caz și exemple).		
Recapitulare (revizuire a celor mai importante subiecte abordate de curs, ghid de examinare).		

Bibliografie

1. B. Stroustrup. *The C++ Programming Language*, Addison Wesley, 1998.
2. Bruce Eckel. *Thinking in C++*, Prentice Hall, 1995.
3. S. Meyers. *Effective C++: 55 Specific Ways to Improve Your Programs and Designs (3rd Edition)*, Addison-Wesley, 2005.
4. S. Meyers. *More effective C++: 35 New Ways to Improve Your Programs and Designs*, Addison-Wesley, 1995.
5. B. Stroustrup. *A Tour of C++*, Addison-Wesley, 2013.
6. C++ reference (<http://en.cppreference.com/w/>).
7. Qt Documentation (<http://doc.qt.io/qt-5/>).
8. E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Longman Publishing, 1995.

8.2 Laborator	Metode de predare - învățare	Observații
Instalarea unui compilator C++, IDE. Noțiuni de bază C/C++.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Probleme introductive (în C).	<ul style="list-style-type: none"> • Explicația • Conversația 	
Programare modulară	<ul style="list-style-type: none"> • Explicația • Conversația 	
Clase și obiecte în C++. Constructorul de copiere, operatorul de asignare, destructori.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Moștenire. Suprascrierea metodelor.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Moștenire și polimorfism, funcții virtuale.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Test de laborator.	<ul style="list-style-type: none"> • Evaluare 	
Containeri STL, iteratori, algoritmi.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Tratarea excepțiilor.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Fișiere. Testarea programelor.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Interfețe grafice în Qt I.	<ul style="list-style-type: none"> • Explicația • Conversația 	

Interfețe grafice în Qt II. Semnale și slot-uri Qt.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Șabloane de proiectare.	<ul style="list-style-type: none"> • Explicația • Conversația 	
Test de laborator.	<ul style="list-style-type: none"> • Evaluare 	

Bibliografie

1. B. Stroustrup. *The C++ Programming Language*, Addison Wesley, 1998.
2. Bruce Eckel. *Thinking in C++*, Prentice Hall, 1995.
3. A. Alexandrescu. *Programarea modernă în C++: Programare generică și modele de proiectare aplicate*, Editura Teora, 2002.
4. S. Meyers. *Effective C++: 55 Specific Ways to Improve Your Programs and Designs (3rd Edition)*, Addison-Wesley, 2005.
5. S. Meyers. *More effective C++: 35 New Ways to Improve Your Programs and Designs*, Addison-Wesley, 1995.
6. B. Stroustrup. *A Tour of C++*, Addison-Wesley, 2013.
7. C++ reference (<http://en.cppreference.com/w/>).
8. Qt Documentation (<http://doc.qt.io/qt-5/>).
9. E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Longman Publishing, 1995.

8.3 Seminar	Metode de predare - învățare	Observații
Probleme simple în C, funcții, structuri, tablouri, tipul de date enumerare (<i>enum</i>)	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	Pe parcursul seminarului, studenții vor implementa iterativ o aplicație care presupune conversia textului în cod Morse. În primele seminarii, vor scrie clasele pentru a genera unde de bază și a le salva în fișiere .csv, care vor fi convertite în fișiere .wav folosind un script Python pus la dispoziție. Ulterior, vor implementa o arhitectură stratificată (<i>layered architecture</i>) pentru a stoca și reda mesaje.
Programare modulară	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	
Clase și obiecte. Supraîncărcarea operatorilor. Fișiere.	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	
Moștenire. Polimorfism. <i>Templates</i> .	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	
Excepții. Containeri STL, iteratori, algoritmi.	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	
Implementarea pe baza diagramelor de clasă. Șabloane de proiectare (Command și Filter).	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	
Interfețe grafice utilizator.	<ul style="list-style-type: none"> • Expunerea interactivă • Explicația • Conversația • Demonstrația didactică 	

Bibliografie

1. B. Stroustrup. *The C++ Programming Language*, Addison Wesley, 1998.
2. Bruce Eckel. *Thinking in C++*, Prentice Hall, 1995.
3. A. Alexandrescu. *Programarea modernă în C++: Programare generică și modele de proiectare aplicate*, Editura Teora, 2002.
4. S. Meyers. *Effective C++: 55 Specific Ways to Improve Your Programs and Designs (3rd Edition)*, Addison-Wesley, 2005.
5. S. Meyers. *More effective C++: 35 New Ways to Improve Your Programs and Designs*, Addison-Wesley, 1995.
6. B. Stroustrup. *A Tour of C++*, Addison-Wesley, 2013.
7. C++ reference (<http://en.cppreference.com/w/>).
8. Qt Documentation (<http://doc.qt.io/qt-5/>).
9. E. Gamma, R. Helm, R. Johnson, J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Longman Publishing, 1995.

9. Evaluare

Tip activitate	9.1 Criterii de evaluare ⁴	9.2 Metode de evaluare ⁵	9.3 Pondere din nota finală
9.4 Curs	Corectitudinea și completitudinea cunoștințelor acumulate și capacitatea de a proiecta și implementa programe C++ corecte.	Examen scris (sesiunea de examene)	60% din nota finală
9.5 Seminar/laborator	Abilitatea de a proiecta, implementa, testa și depana un program C++.	Evaluare practică. Săptămâna a 7-a a semestrului.	20% din nota finală
	Abilitatea de a proiecta, implementa, testa și depana un program C++ cu interfață grafică.	Evaluare practică. Săptămâna a 14-a a semestrului.	20% din nota finală
9.6 Standard minim de promovare			
<ul style="list-style-type: none"> • Studenții trebuie să demonstreze că au dobândit un nivel acceptabil de cunoaștere și înțelegere a conceptelor fundamentale predate la curs, că sunt capabili să utilizeze aceste cunoștințe într-o formă coerentă, că au capacitatea de a stabili anumite conexiuni și de a folosi cunoștințele în rezolvarea unor probleme de mică/medie amploare folosind programarea orientată pe obiecte în C++. • Promovarea examenului este condiționată de obținerea unei note minime de 5 atât la testul practic de laborator, cât și la examenul scris. • Examenul scris este organizat în două părți: o secțiune teoretică și o secțiune practică. Studenții trebuie să obțină o cel puțin nota 5 la ambele părți. • Prezența este obligatorie la 5 ședințe de seminar și 12 ședințe de laborator. 			

10. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)⁶

	<input type="radio"/>	Eticheta generală pentru Dezvoltare durabilă						
								
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

⁴ Criteriile de evaluare trebuie să reflecte direct rezultatele învățării vizate la nivel de program de studii, respectiv la nivel de disciplină. Mai concret, se evaluează achizițiile de învățare menționate în rezultatele anticipate ale învățării.

⁵ Se recomandă stabilirea atât a metodelor de evaluare finală, cât și a strategiei de evaluare pe parcurs.

⁶ Selectați o singură etichetă, cea care, în conformitate cu [Procedura de aplicare a etichetelor ODD în procesul academic](#), se potrivește cel mai bine disciplinei. Dacă disciplina tratează tema dezvoltării durabile la modul general (de ex. prin prezentarea/introducerea cadrului general al dezvoltării durabile etc.) atunci se poate alocă eticheta generală de Dezvoltare Durabilă. Dacă niciuna dintre etichete nu descrie disciplina, selectați ultima opțiune: „Nu se aplică nici o etichetă”.

								Nu se aplică nici o etichetă
								

Data completării:

22.05.2026

Semnătura titularului de curs

Lect.Dr.Ing. Diana-Laura Borza

Semnătura titularului de seminar

Lect.Dr.Ing. Diana-Laura Borza

Data avizării în departament:

...

Semnătura directorului de departament

.....