FIȘA DISCIPLINEI

1. Date despre program

1.1 Instituția de învățământ superior	**Universitatea Națională de Știință și Tehnologie POLITEHNICA din București**
1.2 Facultatea	**Științe Aplicate**
1.3 Departamentul	
1.4 Domeniul de studii universitare	Științe inginerești aplicate
1.5 Programul de studii universitare	Matematică și informatică aplicată în inginerie
1.6 Ciclul de studii universitare	Licență
1.7 Limba de predare	Română
1.8 Locația geografică de desfășurare a studiilor	București

2. Date despre disciplină

2.1 Denumirea disciplinei/ Course title (ro) (en)	**Programarea calculatoarelor si limbaje de programare 2 Computers programming and programming languages 2**						
2.2 Titularul/ii activităților de curs			Prof. dr. ing. Andreea Udrea				
2.3 Titularul/ii activităților de seminar / laborator/proiect			as. drd. Stefan Mihai				
2.4 Anul de studiu	1	2.5 Semestrul/	II	2.6. Tipul de evaluare	E	2.7 Statutul disciplinei	Ob
2.8 Categoria formativă		DF	2.9 Codul disciplinei				

3. Timpul total (ore pe semestru al activităților didactice

3.1 Număr de ore pe săptămână	4	Din care: 3.2 curs	2	3.3 seminar/laborator/proiect	2
3.4 Total ore din planul de învățământ	56	Din care: 3.5 curs/	28	3.6 seminar/laborator/proiect	28

Distribuția fondului de timp:	ore
Studiul după manual, suport de curs, bibliografie și notițe Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate Pregătire seminarii/ laboratoare/proiecte, teme, referate, portofolii și eseuri	40
Tutorat	0
Examinări	4
Alte activități (dacă există):	X

3.7 Total ore studiu individual	**44**
3.8 Total ore pe semestru	**100**
3.9 Numărul de credite	**4**

4. Precondiții (acolo unde este cazul)

4.1 de curriculum	Parcurgerea/promovare **Programarea Calculatoarelor si Limbaje de Programare 1.**
4.2 de rezultate ale învățării	Stapanirea notiunilor: tipuri de date, variabile si operatori; instructiuni conditionale si repetitive; functii si recursivitate; alocarea dinamica a memoriei; structuri, uniuni si enumerari si utilizarea lor in aplicatii implemenatte in limbaj C.

5. Condiții necesare pentru desfășurarea optimă a activităților didactice (acolo unde este cazul)/

5.1 de desfășurare a cursului	Cursul se va desfășura într-o sală dotată cu videoproiector și computer.
5.2 de desfășurare a seminarului/laboratorului/ proiectului	Laboratorul se va desfășura într-o sală cu dotare specifică, care trebuie să includă: PC-uri cu conexiune la internet si IDE C++

6. Obiectiv general

Cursul tratează conceptele fundamentale ale programării orientate pe obiecte: abstractizarea și încapsularea datelor, polimorfismul și moștenirea. Sunt predate mecanismele limbajului C++ care sunt folosite pentru implementarea acestor concepte: clase, obiecte, date și metode; funcții și clase prietene; supradefinirea operatorii și funcțiilor, în general; agregarea și derivarea; redefinirea și supraîncărcarea funcțiilor (funcțiile virtuale); interfete; clase și funcții șablon; clase container și elementele de bază din biblioteca standard C++ (string, fișiere, STL). Obiectivele cursului sunt familiarizarea studenților cu rezolvarea de probleme folosind abordarea pe obiecte, mai degrabă decât folosind proceduri; implementarea și testarea de aplicații orientate pe obiecte. Aplicațiile vin sa completeze noțiunile dobândite la curs și să creeze deprinderi corecte în dezvoltarea aplicațiilor obiectuale. Temele laboratoarelor impun rezolvarea unor probleme concrete cu obiectivul de a fundamenta cunoștințele acumulate în timpul cursurilor.

7. Rezultatele învățării

Cunoștințe	Identifică și descrie concepte, principii și metode de bază din matematică, fizică, chimie, desen tehnic, economie și informatică. Explică și interpretează rezultate teoretice și experimentale din matematică, fizică, chimie, economie, desen tehnic și informatică. Descrie, identifică, sumarizează, prelucrează, concepte și noțiuni elementare referitoare la principii, legi, noțiuni de bază din domeniul științelor fundamentale, analizează și prelucrează modul lor de aplicare în probleme concrete din programului de studii. Descrie, identifică, sumarizează concepte și noțiuni inginerești și modul lor de aplicare în probleme concrete de uz general specifice programului de studii.

Abilități	Operează cu concepte, principii și metode de bază din matematică, fizică, chimie, desen tehnic, economie și informatică. Utilizează metode fundamentale, explică, utilizează, combină, analizează, noțiuni fundamentale, din domeniul științelor fundamentale pentru a implementa, modela și simula fenomene și sisteme specifice domeniului studiat. Utilizează metode și instrumente specifice pentru studiul, analiza, sinteza și realizarea sistemelor și echipamentelor specifice programului de studii. Proiectează, măsoară, evaluează performanțele, diagnostichează și depanează blocuri funcționale de complexitate mică/medie, folosind medii de modelare și simulare dedicate. Proiectează experimente și sisteme inginerești funcționale de complexitate mică/medie specifice.
Responsabilitate și autonomie	Aplică valorile eticii și deontologiei profesiei de inginer. Interpretează legi și principii ale științelor fundamentale ce stau la baza fenomenelor și aparatelor din domeniul de studii. Practică raționamentul logic, evaluarea și autoevaluare în luarea deciziilor. Comunică efficient despre activitățile de inginerie cu o gamă largă de public. Este angajat în învățarea pe tot parcursul vieții pentru dobândirea și implementarea cunoștințelor, după cum este necesar, folosind strategii de învățare adecvate. Selectează și analizează surse bibliografice. Produce software și îl adaptează continuu la noile tehnologii și cerințe de piață. Demonstrează autonomie în învățare Își asumă în mod responsabil sarcinile profesionale și respectă normele de etică și deontologie profesională. Lucrează eficient ca membru în echipă sau lider al acesteia.

8. Metode de predare

Predarea acestei discipline va fi centrată pe student, având ca scop dezvoltarea treptată a competențelor de programare, printr-un echilibru între transmiterea de cunoștințe teoretice și activități practice aplicative.
Procesul didactic va utiliza atât metode expozitive (prelegerea, expunerea de concepte fundamentale de programare și structuri de date), cât și metode interactive și practice, precum: demonstrații de cod, rezolvarea de probleme și exerciții individuale. Pentru situațiile în care se constată dificultăți de înțelegere sau rămâneri în urmă, se vor realiza sesiuni suplimentare de exerciții si exemple explicate pas cu pas.
La începutul fiecărui curs se va face recapitularea noțiunilor de programare din capitolele anterioare, cu accent pe conceptele predate anterior, pentru a asigura continuitatea în învățare. Prezentările vor fi susținute cu materiale vizuale (scheme logice, diagrame de flux, fragmente de cod comentat) pentru a facilita înțelegerea abstractizărilor teoretice.
Activitățile practice vor avea un rol central, vizând dezvoltarea gândirii algoritmice și a capacității de a transpune o problemă într-o soluție programatică. Studenții vor participa la exerciții de rezolvare de probleme, la scrierea și testarea de cod, precum și la proiecte individuale sau de echipă.
De asemenea, se va încuraja oferirea și primirea de feedback constructiv pentru a sprijini progresul individual și adaptarea continuă a metodei de predare la nevoile reale ale grupei.

9. Conținuturi

CURS		
Capitolul	**Conținutul**	**Nr. ore**
I	Limbaje compilate vs limbaje interpretate. Spatiul de adresa virtual al unui proces. Recapitulare notiuni de baza legate de structuri, pointeri si alocare dinamica a memoriei, functii si recursiviate.	2
II	Deosebiri intre C si C++: Comentarii. Pozitia declaratiilor de variabile. Operatorul de rezolutie "::". Functii inline. Argumente de functii cu valori implicite. Supradefinirea functiilor. Operatii de intrare/iesire specific C++. Tipul referinta. Functii membre ale unei structuri. Alocare dinamica – new si delete	2
III	Elemente de baza ale programarii obiectuale: Clase (atribute si metode). Obiecte. Constructorii si destructorul unei clase. Ambiguitati la apelul constructorilor cu parametri cu valori implicite. Autoreferinta this. Cuvantul const in contextual obiectelor, parametrilor unei functii si functii membre const.	2
IV	Functii si clase prietene. Supradefinirea operatorilor. Operatorul de atribuire.	2
V	Supradefinirea operatorilor – continuare. Polimorfism ah-hoc. Exceptii.	2
VI	Biblioteca standard C++. Clasa string. Fisiere text si binare - operatii.	2
VII	Reutilizarea codului prin agregare(compunere) si derivare(mostenire). Agregarea – mecanisme automate de apel constructori, operatori de atribuire si destructori. Agregare puternica si slaba.	2
VIII	Derivare - mecanisme automate de apel constructori, operatori de atribuire si destructori. Derivare publica, protected si private. Supradefinirea si redefinirea functiilor. Conversii la atribuirea obiectelor si pointerilor. Mostenire simpla si multipla.	2
IX	Functii virtuale. Polimorfism.	2
X	Mostenire multipla. Clase virtuale - Problema diamantului mortii. Interfete.	2
XI	Functii si clase sablon. Derivare din si de clase sablon.	2
XII	Clase container. Biblioteca standard template (Standard template library – STL)	4
XIII	Notiuni legate de organizarea aplicatiilor complexe. Exemple.	2
	Total:	**28**

Bibliografie:

1.
2. B. Eckel,Thinking in C++, Second edition, Prentice Hall, 2000
3. W. Savitch, K. Mock, Absolute C++, Student Value Edition (5th Edition), Pearson Education, 2013
4. S. Meyers Effective Modern C++, O'Reilly, 2014
5. G. L. McDoWell, Cracking the coding interview – 6th Edition, CareerCup, 2015
6. S. Meyers, Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14, O'Reilly Media Inc., 2014
7. M. Bancila, Modern C++ Programming Cookbook: Master C++ core language and standard library features, with over 100 recipes, updated to C++20, Second Edition, Packt Publishing, 2020

LABORATOR/ SEMINAR/PROIECT		
Nr. crt.	**Conținutul**	**Nr. ore**
1.	Probleme simple: recursivitate, alocare dinamica a memoriei si structuri	2
2.	Implementarea unei structuri cu funcții membre; unele funcții aflandu-se în relație de supradefinire	2
3.	Implementarea unei clase cu constructori, destructor, funcții membre. Împărțire în header si cpp. Declarare de obiecte și apel metode.	2
4.	Supradefinirea operatorilor pentru o clasa (ex. Număr rațional, Număr complex)	2
5.	Supradefinirea operatorilor pentru o clasa (ex. Vector, Matrice, Polinom, Mulțime).	2
6.	Aplicație în care se manipulează obiecte de tip string și fișiere.	2
7.	Implementarea unei aplicații cu clase agregate.	2
8.	Implementarea unei aplicații cu clase derivate.	2
9.	Implementarea unei ierarhii de clase și utilizarea de funcții virtuale.	2
10.	Implementarea unei ierarhii de clase de tip “death diamond” si utilizarea de clase si funcții virtuale, respectiv, interfețe.	2
11.	Implementarea unei clase sablon cu supradefinire de operatori (ex. Vector, Matrice, Polinom, Mulțime).	2
12.	Implementarea unei aplicații care folosește clasa vector din STL	2
13.	Implementarea unei aplicații care folosește clasele set, multiset, map si multimap din STL	4
	Total:	**28**

Bibliografie:

1.
2. B. Eckel,Thinking in C++, Second edition, Prentice Hall, 2000
3. W. Savitch, K. Mock, Absolute C++, Student Value Edition (5th Edition), Pearson Education, 2013
4. S. Meyers Effective Modern C++, O’Reilly, 2014
5. G. L. McDoWell, Cracking the coding interview – 6th Edition, CareerCup, 2015
6. S. Meyers, Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14, O’Reilly Media Inc., 2014
7. M. Bancila, Modern C++ Programming Cookbook: Master C++ core language and standard library features, with over 100 recipes, updated to C++20, Second Edition, Packt Publishing, 2020

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	Evaluare elemente practice – probleme	Verificari intermediare	20%
	Evaluare elemente practice si teoretice – probleme (25%) si grile(25%)	Examen	50%
10.5 Seminar/laborator/proiect	Evaluare elemente practice – probleme Utilizare noțiuni teoretice în aplicații practice	Rezolvare de probleme (cu notare la fiecare laborator si la final de semestru)	30%
10.6 Condiții de promovare			

Obținerea a 50% din punctajul total.

Data completării	Titular de curs	Titular(ii) de aplicații
	Andreea Udrea	Stefan Mihai
Data avizării în departament	Director de departament	
Data aprobării în Consiliul Facultății 26.09.2025	Decan Alina Claudia PETRESCU-NIȚĂ	