

FIȘA DISCIPLINEI¹

1. Date despre program

1.1 Instituția de învățământ superior	Universitatea Politehnică Timișoara
1.2 Facultatea ² / Departamentul ³	Facultatea de Inginerie Hunedoara / Inginerie Electrică și Informatică Industrială
1.3 Catedra	—
1.4 Domeniul de studii (denumire/cod ⁴)	INGINERIE ELECTRICĂ / 90
1.5 Ciclul de studii	Licenta
1.6 Programul de studii (denumire/cod/calificarea)	INGINERIE ELECTRICĂ ȘI CALCULATOARE / 60 / Inginer

2. Date despre disciplină

2.1 Denumirea disciplinei/Categoria formativă ⁵	Programare orientată pe obiecte / DS						
2.2 Titularul activităților de curs	Prof.dr.ing. Pănoiu Manuela						
2.3 Titularul activităților aplicative ⁶	Șef lucr. Dr. Ing. Ghiormez Loredana						
2.4 Anul de studii ⁷	II	2.5 Semestrul	II	2.6 Tipul de evaluare	E	2.7 Regimul disciplinei ⁸	DI

3. Timp total estimat - ore pe semestru: activități didactice directe (asistate integral sau asistate parțial) și activități de pregătire individuală (neasistate) ⁹

3.1 Număr de ore asistate integral/săptămână	4 , format din:	3.2 ore curs	2	3.3 ore seminar/laborator/proiect	2
3.1* Număr total de ore asistate integral/sem.	56 , format din:	3.2* ore curs	28	3.3* ore seminar/laborator/proiect	28
3.4 Număr de ore asistate parțial/săptămână	, format din:	3.5 ore practică		3.6 ore elaborare proiect de diplomă	
3.4* Număr total de ore asistate parțial/semestru	, format din:	3.5* ore practică		3.6* ore elaborare proiect de diplomă	
3.7 Număr de ore activități neasistate/săptămână	3,14 , format din:	ore documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren			1
		ore studiu individual după manual, suport de curs, bibliografie și notițe			1
		ore pregătire seminarii/laboratoare, elaborare teme de casă și referate, portofolii și eseuri			1,14
3.7* Număr total de ore activități neasistate/semestru	44 , format din:	ore documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren			14
		ore studiu individual după manual, suport de curs, bibliografie și notițe			14
		ore pregătire seminarii/laboratoare, elaborare teme de casă și referate, portofolii și eseuri			16
3.8 Total ore/săptămână ¹⁰	7,14				
3.8* Total ore/semestru	100				
3.9 Număr de credite	4				

4. Precondiții (acolo unde este cazul)

4.1 de curriculum	• Programarea calculatoarelor și limbaje de programare 1, Programarea calculatoarelor și limbaje de programare 2
4.2 de competențe	•

¹ Formularul corespunde Fișei Disciplinei promovată prin OMECTS 5703/18.12.2011 și cerințelor Standardelor specifice ARACIS valabile începând cu 01.10.2017.

² Se înscrie numele facultății care gestionează programul de studiu căruia îi aparține disciplina.

³ Se înscrie numele departamentului căruia i-a fost încredințată susținerea disciplinei și de care aparține titularul cursului.

⁴ Se înscrie codul prevăzut în HG nr.140/16.03.2017 sau în HG similare actualizate anual.

⁵ Disciplina se încadrează potrivit planului de învățământ în una dintre următoarele categorii formative: disciplină fundamentală (DF), disciplină de domeniu (DD), disciplină de specialitate (DS) sau disciplină complementară (DC).

⁶ Prin activități aplicative se înțeleg activitățile de: seminar (S) / laborator (L) / proiect (P) / practică (Pr).

⁷ Anul de studii în care este prevăzută disciplina în planul de învățământ.

⁸ Disciplina poate avea unul din următoarele regimuri: disciplină impusă (DI), disciplină opțională (DO) sau disciplină facultativă (Df).

⁹ Numărul de ore de la rubricile 3.1*, 3.2*,...,3.8* se obțin prin înmulțirea cu 14 (săptămâni) a numărului de ore din rubricile 3.1, 3.2,..., 3.8. Informațiile din rubricile 3.1, 3.4 și 3.7 sunt chei de verificare folosite de ARACIS sub forma: (3.1)+(3.4) ≥ 28 ore/săpt. și (3.8) ≤ 40 ore/săpt.

¹⁰ Numărul total de ore / săptămână se obține prin însumarea numărului de ore de la punctele 3.1, 3.4 și 3.7.

5. Condiții (acolo unde este cazul)

5.1 de desfășurare a cursului	- Sală de curs echipată cu videoproiector și conexiune la Internet. - Studenții nu se vor prezenta la prelegeri cu telefoanele mobile deschise.
5.2 de desfășurare a activităților practice	- Sală de laborator echipată cu videoproiector și computere. - Studenții nu se vor prezenta la activitățile practice cu telefoanele mobile deschise.

6. Competențe la formarea cărora contribuie disciplina

Competențe specifice	C 2. C 2.1. Descrierea funcționării și structurii sistemelor de calcul și a aplicațiilor lor în ingineria electrică folosind cunoștințele referitoare la limbajele, mediile și tehnologiile de programare și la instrumente specifice (algoritmi, scheme, modele, protocoale etc.); C 2.2. Explicarea și interpretarea pachetelor de programe pentru proiectarea și optimizarea sistemelor electrice reprezentative; C 2.3. Rezolvarea de probleme uzuale din domeniul ingineriei electrice folosind pachete de programe dedicate și mijloace de proiectare asistată de calculator (CAD) adecvate; C 2.4. Evaluarea rezultatelor obținute în urma utilizării pachetelor de programe și a mijloacelor de proiectare asistată de calculator (CAD) în rezolvarea problemelor din domeniul ingineriei electrice; C 2.5. Transpunerea problemelor din ingineria electrică în programe de calculator. C 6. C 6.1. Descrierea structurii sistemelor informatice și a modalității de accesare distribuită a resurselor; C 6.2. Identificarea și interpretarea corectă a erorilor semnalate în sistem; C 6.3. Instalarea, configurarea și întreținerea aplicațiilor software specifice ingineriei electrice; C 6.4. Monitorizarea funcționării corecte a sistemului specific și identificarea anomaliilor de funcționare a aplicațiilor software; C 6.5. Proiectarea sistemelor informatice aferente aplicațiilor specifice ingineriei electrice. •
Competențele profesionale în care se înscriu competențele specifice	- C 2. Operarea cu concepte fundamentale din știința calculatoarelor și tehnologia informației. - C 6. Configurarea, realizarea, testarea, exploatarea și întreținerea sistemelor informatice specifice domeniului ingineriei electrice.
Competențele transversale în care se înscriu competențele specifice	•

7. Obiectivele disciplinei (asociate competențelor de la punctul 6)

7.1 Obiectivul general al disciplinei	Scopul acestei discipline îl reprezintă însușirea conceptelor specifice unui limbaj de programare orientat pe obiecte, cum ar fi: tip abstract de date, clasă, obiect, încapsulare, moștenire, derivare și polimorfism.
7.2 Obiectivele specifice	Obiectivele specifice ale acestei discipline sunt: - dezvoltarea deprinderilor de a rezolva probleme de dimensiuni mici și medii într-o manieră orientată pe obiecte; - dezvoltarea deprinderilor de a evidenția diferența între proiectarea procedurală și proiectarea orientată pe obiecte; - dezvoltarea deprinderilor de a să înțeleagă rolul moștenirii, polimorfismului, legării dinamice și a structurilor generice în dezvoltarea unor programe reutilizabile; - dezvoltarea deprinderilor de a implementa programe în limbajul de programare

	C++ de dimensiuni mici/medii; dezvoltarea deprinderilor de a utiliza clase scrise de alți programatori.
--	--

8. Conținuturi¹¹

8.1 Curs	Număr de ore	Metode de predare ¹²
1. Introducere în programarea orientată pe obiect 1.1. Declararea și implementarea unei clase. Pointerul this 1.2. Protejarea informațiilor din cadrul obiectelor. Încapsulare 1.3. Funcții membre inline	2	Studentii au acces la curs în format electronic. Se vor utiliza atât prezentări interactive cât și tradiționale. Se vor folosi: problematizarea, studiu de caz, conversația.
2. Crearea, instanțierea, inițializarea și distrugerea obiectelor 2.1. Liste de instanțieri 2.2. Constructori. Tipuri de constructori 2.3. Destructorii 2.4. Manevrarea dinamică a obiectelor	4	
3. Elemente caracteristice claselor 3.1. Date și funcții membre statice 3.2. Obiecte constante. Funcții membre constant 3.3. Funcții și clase prietene 3.4. Clase imbricate. Clase locale 3.5. Transmiterea obiectelor către funcții. Returnarea obiectelor de către funcții	4	
4. Moștenire și polimorfism 4.1. Derivarea claselor. Reutilizarea funcțiilor din clase derivate 4.2. Polimorfismul. Supraîncărcarea și supradefinirea funcțiilor. Funcții virtuale 4.3. Protecția informației într-o ierarhie de clase 4.4. Clase virtuale. Clase abstracte 4.5. Supraîncărcarea operatorilor	6	
5. Tratarea excepțiilor. Sistemul de intrare/ieșire din C++ 5.1. Declararea excepțiilor. Ierarhii de excepții 5.2. Ierarhia de clase corespunzătoare stream-urilor 5.3. Stream-uri predefinite în C++ 5.4. Redirecționări. Supraîncărcarea operatorilor << și >> 5.5. Operații de intrare/ieșire cu fișiere în C++	6	
6. Programare generică în C++. Biblioteca standard C++ 6.1. Funcții și clase generice în C++ 6.2. Biblioteca standard C++ 6.2.1. Containere 6.2.2. Iteratori 6.2.3. Algoritmi 6.2.4. Functori. Clase de functori	6	
Bibliografie¹³ 1. Manuela Pănoiu, Anca-Elena Iordan, Programare orientată pe obiecte în C++, Editura Politehnica, Timișoara, 2014 2. Liviu Negrescu, Limbajele C și C++ pentru începători, Volumul II – Limbajul C++, Editura Alabastră, Cluj-Napoca, 2006 3. Scott Meyers, STL – Biblioteca programatorului, Editura Teora, București, 2002 4. Constantin Gălățan, C++ - Introducere în Standard Template Library, Editura All, București, 2008 5. Johan Thelin, Foundations of Qt Development, APress, 2007		
8.2 Activități aplicative ¹⁴	Număr de ore	Metode de predare

¹¹ Se detaliază toate activitățile didactice prevăzute prin planul de învățământ (tematicile prelegerilor și ale seminariilor, lista lucrărilor de laborator, conținuturile etapelor de elaborare a proiectelor, tematica fiecărui stagi de practică). Titlurile lucrărilor de laborator care se efectuează pe standuri vor fi însoțite de notația „(*)”.

¹² Prezentarea metodelor de predare va include și folosirea noilor tehnologii (e-mail, pagină personalizată de web, resurse în format electronic etc.).

¹³ Cel puțin un titlu trebuie să aparțină colectivului disciplinei iar cel puțin un titlu trebuie să se refere la o lucrare de referință pentru disciplină, de circulație națională și internațională, existentă în biblioteca UPT.

1. Analiza, proiectarea și implementarea de aplicații care utilizează tipuri de date abstracte definite prin intermediul structurilor.	4	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
2. Analiza, proiectarea și implementarea de aplicații care utilizează tipuri de date abstracte definite prin intermediul claselor.		
3. Analiza, proiectarea și implementarea de aplicații care utilizează constructori și destructori.	2	
4. Analiza, proiectarea și implementarea de aplicații care utilizează date și funcții statice. Funcții prietene	2	
5. Analiza, proiectarea și implementarea de aplicații care utilizează compunerea de obiecte.	4	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
6. Analiza, proiectarea și implementarea de aplicații care utilizează conceptul de moștenire.		
7. Analiza, proiectarea și implementarea de aplicații care utilizează funcții virtuale pure. Clase abstracte	2	
8. Analiza, proiectarea și implementarea de aplicații care utilizează supraîncărcarea operatorilor.	4	
9. Analiza, proiectarea și implementarea de aplicații care utilizează ierarhii de clase pentru operații de intrare/ieșire.	2	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
10. Analiza, proiectarea și implementarea de aplicații prin reutilizarea codului sursă cu ajutorul claselor șablon.	2	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
11. Analiza, proiectarea și implementarea de aplicații prin reutilizarea codului sursă cu ajutorul containerelor secvențiale STL.	6	Se va utiliza exercițiul la tablă și implementarea programului utilizând calculatorul.
Bibliografie ¹⁵ 1. Anca-Elena Iordan, Manuela Pănoiu, Programare orientată pe obiecte – C++, Editura Mirton, Timișoara, 2007 2. Valeriu Iorga, Cristian Opincaru, Corina Stratan, Alexandru Chiriță, Structuri de date și algoritmi. Aplicații în C++ folosind STL, Editura Polirom, Iași, 2005 3. Jasmin Blanchette, Mark Summerfield, C++ GUI Programming with Qt 4, 2008 4. Nicolai Josuttis, The C++ Standard Library, Pearson Education, 2012		

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

Disciplina vine în întâmpinarea așteptărilor angajatorilor reprezentativi din domeniul aferent programului prin conținutul orelor de curs și laborator.

10. Evaluare

Tip activitate	10.1 Criterii de evaluare ¹⁶	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	- corectitudinea și completitudinea cunoștințelor asimilate; Studenții trebuie să fie capabili să implementeze programe C++ - criterii ce vizează aspecte atitudinale: interesul pentru studiu individual și dezvoltare profesională	Examen on-line pe campusul virtual cu întrebări cu caracter aplicativ, inclusiv programe C++; studenții pot folosi bibliografia, dar fără conexiune la internet	0,66

¹⁴ Tipurile de activități aplicative sunt cele precizate în nota de subsol 5. Dacă disciplina conține mai multe tipuri de activități aplicative atunci ele se trec consecutiv în liniile tabelului de mai jos. Tipul activității se va înscrie într-o linie distinctă sub forma: „Seminar:”, „Laborator:”, „Proiect:” și/sau „Practică:”.

¹⁵ Cel puțin un titlu trebuie să aparțină colectivului disciplinei.

¹⁶ Fișele disciplinelor trebuie să conțină procedura de evaluare a disciplinei cu precizarea criteriilor, a metodelor și a formelor de evaluare, precum și cu precizarea ponderilor atribuite acestora în nota finală. Criteriile de evaluare se formulează în mod distinct pentru fiecare activitate prevăzută în planul de învățământ (curs, seminar, laborator, proiect). Ele se vor referi și la formele de verificare pe parcurs (teme de casă, referate ș.a.)

10.5 Activități aplicative	S:		
	L: Abilități în analiza, proiectarea și implementarea aplicațiilor de laborator	Oral – aplicații utilizând calculatorul	0,34
	P¹⁷:		
	Pr:		
10.6 Standard minim de performanță (se prezintă cunoștințele minim necesare pentru promovarea disciplinei și modul în care se verifică stăpânirea lor¹⁸)			
- Nota de promovare se obține în condițiile obținerii a minim jumătate din punctajul total. - La finalul cursului, respectiv al laboratorului, studenții trebuie să aibă cunoștințe teoretice și abilități de cercetare, strict necesare viitorilor specialiști dovedind competențe în implementarea programelor pe baza conceptului de programare orientată pe obiecte în C++ utilizând biblioteca STL			

Data completării

10.09.2024

**Director de departament
(semnătura)**

.....


**Titular de curs
(semnătura)**

.....


Data avizării în Consiliul Facultății¹⁹

17.09.2024

**Titular activități aplicative
(semnătura)**

.....


**Decan
(semnătura)**

.....


¹⁷ În cazul când proiectul nu este o disciplină distinctă, în această rubrică se va preciza și modul în care rezultatul evaluării proiectului condiționează admiterea studentului la evaluarea finală din cadrul disciplinei.

¹⁸ Nu se va explica cum se acorda nota de promovare.

¹⁹ Avizarea este precedată de discutarea punctului de vedere al board-ului de care aparține programul de studii cu privire la fișa disciplinei.