FIŞA DISCIPLINEI

1. Date despre program

1.1 Instituţia de învăţământ superior	Universitatea Politehnica Timișoara
1.2 Facultatea[1] / Departamentul[2]	Electronică, Telecomunicații și Tehnologii Informaționale / Comunicații
1.3 Domeniul de studii (denumire/cod [3])	Inginerie electronică, telecomunicații și tehnologii informaționale / 20.20.100.10
1.4 Ciclul de studii	Licență
1.5 Programul de studii (denumire/cod/calificarea)	Tehnologii și Sisteme de Telecomunicații / 20.20.100.10/ Tehnologii și Sisteme de Telecomunicații

2. Date despre disciplină

2.1a Denumirea disciplinei/Categoria formativă[4]	Limbaje de programare 2 / DF						
2.1b Denumirea disciplinei în limba engleză	Programming Languages 2						
2.2 Titularul activităţilor de curs	Conf.dr.ing. Bucos Marian						
2.3 Titularul activităţilor aplicative[5]	Conf.dr.ing. Bucos Marian						
2.4 Anul de studii[6]	2	**2.5** Semestrul	1	**2.6** Tipul de evaluare	V	**2.7** Regimul disciplinei[7]	DOB

3. Timp total estimat - ore pe semestru: activități didactice directe (asistate integral sau asistate parțial) și activități de pregătire individuală (neasistate) [8]

3.1 Număr de ore asistate integral/săptămână	4 , format din:	**3.2** ore curs	2	**3.3** ore seminar/laborator/proiect	0/2/0
3.1* Număr total de ore asistate integral/sem.	56 , format din:	**3.2*** ore curs	28	**3.3*** ore seminar/laborator/proiect	0/28/0
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			1.14
		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
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			16
		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			14
3.8 Total ore/săptămână [9]	7.14				
3.8* Total ore/semestru	100				
3.9 Număr de credite	4				

4. Precondiţii (acolo unde este cazul)

4.1 de curriculum	•
4.2 de rezultatele învățării	• Competențe de bază în utilizarea și programarea calculatoarelor.

5. Condiţii (acolo unde este cazul)

5.1 de desfăşurare a cursului	• Sală cu proiector și tablă.
5.2 de desfăşurare a activităţilor practice	• Laborator de informatică cu suficiente calculatoare pentru numărul de studenți.

6. Rezultatele învățării la formarea cărora contribuie disciplina

Cunoștințe	• C1. Studentul/absolventul identifică și descrie concepte, principii și metode de bază din matematică, fizică,

	grafică asistată de calculator, bazele electrotehnicii, limbaje de programare. • C2. Studentul/absolventul explică și interpretează rezultate teoretice și experimentale din matematică, fizică, analiza si sinteza circuitelor, programarea calculatoarelor, si grafica asistata de calculator. • C5.Studentul/absolventul descrie, identifică, sumarizează concepte și metode elementare privitoare la arhitectura sistemelor de calcul, microcontrolere, limbaje și tehnici de programare și modul lor de aplicare în probleme concrete.
Abilități	• A1.Studentul/absolventul operează cu concepte, principii și metode de bază din matematică, fizică, grafică asistată de calculator, bazele electrotehnicii, limbaje de programare. • A17. Studentul/absolventul specifică cerințe, elaborează programe în limbaje de programare de uz general (C, etc.) și /sau obiect-orientate (C++, Java, etc.), execută, depanează și interpretează rezultatele programelor realizate în vederea rezolvării unei probleme concrete. • A19. Studentul/absolventul elaborează și rezolvă exerciții practice, lucrări de laborator și probleme aplicative, demonstrând capacitatea de integrare a noțiunilor teoretice.
Responsabilitate și autonomie	• RA2. Studentul/absolventul practică raționamentul logic, evaluarea și autoevaluare în luarea deciziilor. • RA3. Studentul/absolventul comunică eficient despre activitățile de inginerie cu o gamă largă de public. • RA6. Studentul/absolventul lucrează eficient ca membru în echipă sau lider al acesteia. • RA10. Studentul/absolventul manifestă capacitatea de autoorganizare și de gestionare a timpului de studiu, respectând cerințele și termenele activităților academice.

7. Obiectivele disciplinei (asociate rezultatelor învățării de la punctul 6)

- Disciplina are ca scop consolidarea și extinderea cunoștințelor studenților privind conceptele fundamentale ale limbajelor de programare orientate pe obiecte (clasă, obiect, abstractizare, încapsulare, moștenire, polimorfism), modelarea problemelor reale și găsirea de soluții prin implementarea de cod specific pentru prelucrarea datelor și aplicații de rețea. Prin parcurgerea acestei discipline, studenții vor înțelege și aplica tipuri de date, structuri de control, funcții, structuri de date integrând atât paradigma orientată pe obiect, cât și elemente funcționale specifice moderne.
- Studenții pot proiecta și implementa aplicații orietate pe obiecte folosind principii solide de design, pot testa și depana sistematic, pot gestiona concurența și I/O, pot documenta și livra software de calitate, lucrând eficient individual și în echipă și comunicând clar deciziile tehnice.

8. Conţinuturi[10]

8.1 Curs	Număr de ore	Metode de predare[11]
Limbajul de programare Java. Programarea Orientată pe Obiecte. Platforme Java. Instalarea Java SDK. Compilarea și rularea unui program. Documentarea programelor.	2	Prelegere participativa, problematizare, dezbatere, verificare. Utilizarea resurselor educaționale deschise, consultații prin intermediul platformelor electronice (email, Microsoft Teams).
Utilizarea comentariilor într-un program sursă. Operatorii și precedența lor. Tipuri de date primitive și referință. Declararea variabilelor. Instrucțiuni Java pentru controlul execuției. Tablouri.	2	
Clase Java. Definirea unui clase. Utilizarea modificatorilor. Declararea variabilelor și implementarea metodelor într-o clasă. Instanțierea obiectelor unei clase.	2	
Paradigme de programare orientată pe Obiecte. Clase. Instanțe. Moștenire. Polimorfism. Încapsulare. Diferențieri în implementare.	2	
Ierarhii de clase. Clase și metode abstracte. Crearea și utilizarea interfețelor. Pachete de clase. Arhive Java.	4	
Excepții. Generarea excepțiilor. Categorii de excepții. Interceptarea și tratarea excepțiilor. Definirea de excepții utilizator.	2	
Definirea conceptului de flux de date și clasificarea fluxurilor de date. Ierarhia claselor pentru lucrul cu fluxuri de date. Fluxuri standard de intrare/ieșire. Utilizarea fluxurilor de date.	2	
Operații de intrare/ieșire. Operații cu fișiere. Serializarea datelor. Organizarea aplicațiilor folosind module și pachete.	2	
Colectii de obiecte. Java Collections Framework. Interfete. Implementari. Interfata Collection. Parcurgerea colectiilor.	4	

Interfata List. Interfata Set. Interfata Map.		
Interfețe grafice. Interfața grafica cu utilizatorul. Pachetele awt si swing. Suprafețe de afișare. Gestionarea poziționării. Componente grafice.	2	
Tratarea evenimentelor. Tipuri de evenimente. Interceptoare de evenimente. Tipuri de interceptoare de evenimente.	2	
Lucrul cu baze de date in Java. Baze de date relationale. Java DataBase Connectivity (JDBC). Stabilirea unei conexiuni. Rularea unei comenzi SQL. Manipularea si prelucrarea rezultatelor.	2	

Bibliografie[12] M. Bucos, Programare Orienată pe obiecte, Editura Politehnica, 978-606-554-851-0, 2014
B. Eckel, Thinking in Java, Pearson, 978-0-13-187248-6, 2006
A.B. Downey, C. Mayfield, Think Java, 2nd Edition, O'Reilly Media, 978-149-207-250-8, 2019
D. Andrei, M. Bolog, E. Țundrea, Aplicații web cu suport Java: notițe de curs, Editura Politehnica, 978-606-350-587-4, 2023

8.2 Activităţi aplicative[13]	Număr de ore	Metode de predare
Introducere in POO. Instalare. Lucrul cu interpretorul. Medii integrate de dezvoltare.	2	Expunere, discuție liberă, problematizare, aplicație practică, verificare. Utilizarea resurselor educaționale deschise.
Tipuri de date. Operatori. Expresii. Instrucțiuni. Tipul tablou. Metode.	2	
Clase. Paradigme POO. Clase abstracte. Interfețe.	4	
Excepții. Tratarea excepțiilor.	2	
Operații de intrare/ieșire. Pachete de clase.	4	
Colecții de obiecte.	4	
Serializarea datelor.	4	
Lucrul cu baze de date.	2	
Interfețe grafice. Tratarea evenimentelor.	4	

Bibliografie[14] M. Mocofan, M. Bucos, Programare orientată pe obiecte - activități practice, http://cv.upt.ro, 2025
***, JavaSE Documentation, https://docs.oracle.com/en/java/javase/, 2025
***, Eclipse IDE for Java Developers, https://www.eclipse.org/, 2025
***, MySQL Documentation: MySQL Reference Manuals, http://dev.mysql.com/doc/, 2025

9. Evaluare

Tip activitate	**9.1** Criterii de evaluare[15]	**9.2** Metode de evaluare	**9.3** Pondere din nota finală
9.4 Curs	Cunoașterea noțiunilor și conceptelor fundamentale prezentate la curs și laborator	Evaluare cunoștințelor se realizează prin examen scris distribuit în două părți. Partea teoretică este evaluată prin teste cu itemi de mai multe tipuri, iar partea practică prin itemi de tip grilă în care se dorește evaluarea unei secvențe scurte de cod. Implementarea evaluării se realizează în format electronic prin intermediul platformei Campus Virtual.	50%
9.5 Activităţi aplicative	**S:**		
	L: Aplicarea cunoștințelor pentru rezolvarea unor probleme. Rezolvarea cerințelor.	Evaluare cu ajutorul calculatorului prin rezolvarea unor probleme. Vor fi cel puțin două teste de acest tip. Suplimentar se realizează evaluare scurtă la finalul unor laboratoare prin teste grilă implementate în platforma Campus Virtual.	50%
	P[16]**:**		
	Pr:		

9.6 Standard minim de performanţă (se prezintă cunoștințele minim necesare pentru promovarea disciplinei și modul în care se verifică stăpânirea lor[17])

- Scriere cod fără erori de sintaxă. Se verifică în cadrul testelor practice.
- Alegerea corectă a structurilor de date pentru probleme specifice. Se verifică prin examen și teste practice.
- Implementarea unor secvențe de cod de complexitate scăzută. Se verifică în cadrul testelor practice.
- Recunoașterea părților componente și rolul lor în cadrul unei secvențe de cod de complexitate medie. Se verifică prin examen.

Data completării 15.09.2025	**Titular de curs** **(semnătura)**	**Titular activităţi aplicative** **(semnătura)**
Director de departament **(semnătura)**	**Data avizării în Consiliul Facultăţii**[18] 07.10.2025	**Decan** **(semnătura)**