F I Ş A D I S C I P L I N E I [1)]

1. Date despre program

1.1. Instituţia de învăţământ superior	Universitatea Petrol-Gaze din Ploiești
1.2. Facultatea	Litere și Științe
1.3. Departamentul	Informatică, Tehnologia Informației, Matematică și Fizică
1.4. Domeniul de studii universitare	Informatică
1.5. Ciclul de studii universitare	Licență
1.6. Programul de studii universitare	Informatică

2. Date despre disciplină

2.1. Denumirea disciplinei	Inginerie Software
2.2. Titularul activităţilor de curs	Conf. dr. ing. Monica Vladoiu
2.3. Titularul activităţilor aplicative	Asist. Dr. Elia Dragomir
2.4. Anul de studiu	III
2.5. Semestrul*	5
2.6. Tipul de evaluare	C
2.7. Categoria formativă** / regimul*** disciplinei	F0/ O

*numărul semestrului este conform planului de învăţământ;

** fundamentală = F0; de domeniu = D1; de specialitate = S2; complementară = C3

***obligatorie = O; opţională = A; facultativă = L

3. Timpul total estimat (ore pe semestru al activităţilor didactice)

3.1. Număr de ore pe săptămână	4	din care: 3.2. curs	2	3.3. Seminar/laborator	2
3.4. Total ore din planul de învăţământ	56	din care: 3.5. curs	28	3.6. Seminar/laborator	28

3.7. Distribuţia fondului de timp	ore
Studiu după manual, suport de curs, bibliografie şi notiţe	30
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren	20
Pregătire seminarii/laboratoare, teme, referate, portofolii şi eseuri	44
Tutoriat	-
Examinări	-
Alte activităţi	-

3.7. Total ore studiu individual	69
3.8. Total ore pe semestru	125
3.9. Numărul de credite	5

4. Precondiţii (acolo unde este cazul)

4.1. de curriculum	Programarea Procedurală, Programarea Procedurală Avansată, Algoritmi şi Structuri de Date, Programare Orientată pe Obiecte
4.2. de competenţe	Cunoaşterea programării procedurale (paradigmă şi limbaj, fundamente şi elemente avansate) Cunoaşterea fundamentelor analizei, proiectării şi programării orientate pe obiecte Cunoasterea principalelor structuri de date si a algoritmilor de prelucrare a acestora

5. Condiţii (acolo unde este cazul)

5.1. de desfăşurare a cursului	sală de curs multimedia cu videoproiector şi conexiune la Internet
5.2. de desfăşurare a seminarului/laboratorului	sală de laborator echipată cu reţea de calculatoare şi software corespunzător pentru dezvoltare de aplicaţii în Java

[1)] Adaptare după Ordinul Ministrului educaţiei, cercetării, tineretului şi sportului nr. 5 703/2011 privind implementarea Codului naţional al calificărilor din învăţământul superior, publicat în Monitorul Oficial al României, partea I, nr.880 bis / 13.XII.2011

6. Competenţe specifice acumulate

Competenţe profesionale	• C2.1 Identificarea de metodologii adecvate de dezvoltare a sistemelor software; • C2.2 Identificarea şi explicarea mecanismelor adecvate de specificare a sistemelor software; • C2.3 Utilizarea metodologiilor, mecanismelor de specificare și a mediilor de dezvoltare pentru realizarea aplicațiilor informatice; • C2.4 Utilizarea de criterii și metode adecvate pentru evaluarea aplicațiilor informatice; • C2.5 Realizarea unor proiecte informatice dedicate; • C3.2 Identificarea si explicarea modelelor informatice de baza adecvate domeniului de aplicare; • C3.3 Utilizarea modelelor si instrumentelor informatice si matematice pentru rezolvarea problemelor specifice domeniului de aplicare; • C4.2 Interpretarea de modele matematice şi informatice (formale); • C4.3 Identificarea modelelor si metodelor adecvate pentru rezolvarea unor probleme reale; • C4.5 Încorporarea de modele formale în aplicaţii specifice din diverse domenii.
Competenţe transversale	• CT1. Aplicarea regulilor de muncă organizată şi eficientă, a unor atitudini responsabile faţă de domeniul didactic, ştiinţific şi profesional, în vederea valorificării creative a propriului potenţial, cu respectarea principiilor şi normelor de etică profesională; • CT2. Desfăşurarea eficientă a activităţilor organizate în echipă și dezvoltarea capacităţilor empatice şi de comunicare inter-personală, de relaţionare şi colaborare cu persoane şi grupuri diverse implicate în dezvoltarea şi utilizarea de sisteme software; • CT3. Utilizarea de metode şi tehnici eficiente de învăţare, informare, cercetare şi dezvoltare a capacităţilor de valorificare a cunoştinţelor, dar şi de adaptare la cerinţele unei societăţi dinamice şi în continuă schimbare, precum şi dezvoltarea capacităţii de a comunica eficient şi profesionist atît în limba română, cît şi într-o limbă de circulaţie internaţională, prin însuşirea şi folosirea adecvată a limbajului de specialitate.

7. Obiectivele disciplinei (reieşind din grila competenţelor specifice acumulate)

7.1. Obiectivul general al disciplinei	*Dobîndirea de către studenţi a cunoştinţelor fundamentale privind dezvoltarea de sisteme mari de programe şi aplicarea corespunzătoare a acestora în practica dezvoltării acestor sisteme, alături de deprinderea de a lucra în echipe mici de programatori şi de a prezenta convingător rezultatele efortului de echipă.*
7.2. Obiectivele specifice	*După parcurgerea disciplinei studenţii vor putea să:* • facă diferenţa între dezvoltarea de programe de mici dimensiuni si abordarea cu metode ingineреşti a dezvoltării sistemelor şi aplicaţiilor software complexe, de mari dimensiuni; • rezume corect principalele caracteristici ale unui pachet software „de încredere"; • să descrie şi să aplice în mod adecvat principiile Codului de practică şi etică profesională în Ingineria Programării; • descrie şi să analizeze comparativ principalele modele ale procesului de dezvoltare de software; • descrie principalele caracteristici, particularităţi şi probleme din managementul dezvoltării de sisteme mari de programe; • explice importanţa factorilor umani în ingineria programării; • explice şi să aplice adecvat principiile ingineriei sistemelor şi aplicaţiilor software complexe, ale specificării, proiectării, dezvoltării şi validării lor, în contextul folosirii unor platforme integrate şi sub incidenţa unui management adecvat; • dezvolte pachete software integrate cu structuri de date fundamentale, folosind limbajul Java; • lucreze în cadrul unor echipe mici de programatori pentru a finaliza şi prezenta corespunzător un proiect software atît către management, cît şi către client .

8. Conţinuturi

8.1. Curs	Nr.ore	Metode de predare	Observaţii
1. Ingineria programării (IP). Software bine făcut. Diversitatea sistemelor software. Activităţi şi principii ale IP. Încrederea în sistemele software. Etica în IP.	6	• prelegeri active și	

2. Modele ale procesului software. Modelul cascada. Dezvoltarea incrementala si iterativa. IP orientata pe reutilizare. Prototipizarea. Modelul spirala al lui Boehm. Procesul unificat Rational. Metode si procese de tip Agile (eXtreme Programming, Scrum). Free si Open Source Software.	4	angajante; • supervizare și mentorat "deschise"; • învățarea prin descoperire; • învățare pe grupuri; • învățare bazată pe proiecte şi pe studii de caz; • învățare bazată pe rezolvarea de probleme; • învăţare centrată pe student; • learning by doing; • brainstorming; • învățare hibridă cu folosirea resurselor educaţionale open; • învățare reflectivă etc.	
3. Managementul proiectelor software. Generalitati, particularitati si obiective. Functii si activitati. Factori umani. Structuri manageriale. Productivitatea programatorului. Planificarea si realizarea orarului. Mentenanta si evolutia software-ului. Managementul configuratiilor. Documentatia. Asigurarea calitatii.	4		
4. Proiectarea sistemelor de programe -. Proiectarea programelor si a sistemelor software. Modelarea proiectarii sistemelor software - proiectare orientata pe obiecte, proiectare orientata pe functii. Proiectarea interfetei utilizator.	4		
5. Specificatia sistemelor de programe - Fundamente ale specificarii programelor. Specificarea cerintelor. Validare si prototipizare. Specificarea software-ului (axiomatica, algebrica, bazata pe model)	2		
6. Validarea sistemelor de programare - Verificarea si validarea programelor. Tehnici de testare. Verificare statica. Utilitare pentru testare si depanare	2		
7. Practici de programare, tehnici si medii integrate de dezvoltare.	2		

Bibliografie

1. Braude, E. J., *Software engineering: Modern approaches*, New Jersey, John Wiley & Sons, 2011*
2. Ghezzi, C., *Fundamentals of Software Engineering*, New Jersey, Prentice Hall, 2003*
3. Jacobson, I., *The essence of software engineering: Applying the SEMAT kernel*, Boston, San Francisco, Addison-Wesley, 2013*
4. Jones, C., *Software engineering best practices: Lessons from successful projects in top companies*, New-York, McGraw Hill, 2010*
5. *Jones, P. H., Team Design: A Practitioner's Guide to Collaborative Innovation, Xlibris, 2002**
6. McConell, S., *Code Complete: A practical handbook of software construction*, Microsoft Press, 2004*
7. Sommerville I., *Software Engineering*, Pearson, Boston New York, 2011*
8. Vlădoiu, M., Constantinescu Z., Moise, G., *Ingineria Programării. Fundamente*, Ed. UPG Ploieşti, 2015*
9. Resurse educationale disponibile la http://www.unde.ro/cursuri/IP/ şi http://www.unde.ro/cursuri/OCW/

* Disponibile la biblioteca departamentului iTIMF

8.2. Seminar / laborator/proiect	Nr. ore	Metode de predare	Observaţii
1. Programare orientata pe obiecte - notiuni fundamentale, clase si obiecte, atribute si metode, mostenire si polimorfism. Un exemplu de complexitate medie in C++ si Java. Comparatie. Implementare folosind NetBeans. Operatii de intrare-iesire.	2	• învățarea prin descoperire; • învățare pe grupuri; • învățare bazată pe proiecte, • învățare bazată pe rezolvarea de probleme; • învăţare centrată pe	
2. Pogramare in Java in linie de comanda. Operatii de intrare-iesire.	2		
3, 4. Tablouri in Java. Implementare atit in NetBeans, cit si in linie de comanda.	4		
5. Sortari in java (metoda bulelor, prin insertie, prin selectie).	2		
6. Liste inlantuite in Java. Lista cu header (cu doua capete). Iteratori in Java.	2		
7. Liste inlantuite sortate in Java. Sortarea listelor prin insertie.	2		
8. Stive in Java. Implementare cu tablouri si cu liste inlantuite. Inversarea unui sir de caractere, imperecherea delimitatorilor.	2		

9. Cozi in java - implementare cu tablouri si cu liste inlantuite.	2	student; • learning by doing; • brainstormin; • învățare hibridă; • folosirea resurselor educaţionale open; • învățare reflectivă etc.	
10. Liste dublu inlantuite in Java. Alegere teme proiect.	2		
11. Arbori binari in Java - implementare folosind referinte. Constructie si parcurgere RSD (preordine). Calculul inaltimii. Cautarea unei chei.	2		
12. Arbori binari in Java Constructie si parcurgere SRD (inordine) si SDR (postordine). Verificarea echilibrarii arborelui.	2		
13, 14. Dezvoltarea unei interfete grafice utilizator folosind elemente de design (etichete, butoane, casete de validare, liste de optiuni, zone de text, layout managers etc.). Integrarea unui set de programe pentru a crea o aplicatie Java unica.	4		

Bibliografie

1. Waite M., Lafore R., *Structuri de date şi algoritmi în Java*, Teora, 2000*
2. Lemay L., Cadenhead R., *Java 2 fara profesor in 21 de zile*, Teora, 2000
3. Chan M. C., Griffith S. W., Iasi A. F., *Java - 1001 secrete pentru programatori*, Teora, 2000
4. Sedgewick, R., *Introduction to programming in Java. An interdisciplinary approach*, PearsonEducation Ltd,2014*
5. Resurse educationale disponibile la http://www.unde.ro/cursuri/IP/ şi http://www.unde.ro/cursuri/OCW/

* Disponibile la biblioteca departamentului iTIMF

9. Coroborarea conţinuturilor disciplinei cu aşteptările reprezentanţilor comunităţii epistemice, asociaţilor profesionale şi angajatori reprezentativi din domeniul aferent programului

- Conţinuturile disciplinei corespund cu aşteptările reprezentanţilor comunităţii epistemice, asociaţiilor profesionale şi angajatorilor reprezentativi din domeniul aferent programului, aşa după cum rezultă din prezenta fişă, dar şi din fişa specializării, acestea fiind în concordanţă deplină cu CNCIS şi COR;
- Disciplina de faţă respectă recomandările IEEE/CS şi ACM legate de planul de învăţământ şi de conţinuturile necesare pentru specializarea Informatică/Ştiinţa Calculatoarelor;
- Disciplina de faţă există în planurile de învăţământ al tuturor marilor universităţi din România şi din străinătate.

10. Evaluare

Tip activitate	10.1. Criterii de evaluare	10.2. Metode de evaluare	10.3. Pondere din nota finală
10.4. Curs	Dobîndirea competenţelor profesionale şi transversale specifice disciplinei	*Proiect:* dezvoltare sistem software in echipa mică de programatori, aplicaţie informatică integrată şi documentaţie *Examinare orală* pe teme fundamentale din conţinutul cursului	Documentaţie 40% Aplicaţie informatică 35% Examinare orală 15% Din oficiu 10%
10.5. Seminar/ laborator/proiect			
10.6. Standard minim de performanţă			
• RNCIS: Realizarea și întreținerea unor aplicații informatice pentru rezolvarea unor probleme reale de complexitate medie; realizarea componentelor informatice pentru o aplicaţie dedicată de complexitate medie; modelarea şi rezolvarea unor probleme cu grad mediu de complexitate, folosind cunoştinţe de matematică si informatica; • *Realizarea şi prezentarea proiectului dezvoltat care să conţină minim punctele 1, 2, 3, 4 şi 8-documentaţia utilizator din lista cu conţinutul proiectului prezentată la sfîrşitul capitolului 2, Managementul software*			

Data completării _____21.09.20_____ Semnătura titularului de curs _____ Semnătura titularului de seminar/laborator _____

Data avizării în departament _____ Semnătura directorului de departament _____