UNIVERSITATEA "DUNĂREA DE JOS" DIN GALAȚI
FACULTATEA STIINTA CALCULATOARELOR

SPECIALIZAREA : **CALCULATOARE**

FORMA DE ÎNVĂȚĂMÂNT
ZI
ANUL DE STUDIU: IV
SEMESTRUL : II

FISA DISCIPLINEI

PROGRAMARE LOGICĂ ŞI FUNCȚIONALĂ
COBD405

Număr credite 6

1. Obiectivele disciplinei

Însuşirea cunoştinţelor teoretice din domeniul programării logice şi funcționale. Dezvoltarea cunoştinţelor pentru crearea de aplicații în limbajele Prolog şi ChezScheme, ceea ce implică studierea aspectelor de sintaxă şi de semantică ale limbajelor logice şi funcționale.

2. Rezultatele învățării (*se exprima in obiective măsurabile ce fac subiectul evaluării*)

a. Cunoştinţele generale

Clasificarea limbajelor de programare. Caracteristicile limbajelor de programare. Avantajele limbajelor de programare de nivel înalt. Prezentarea limbajelor procedurale/declarative şi a celor interpretate/compilate. Principiile programării. Structuri de programare. Tipuri şi structuri de date. Taxonomia limbajelor de programare. Elemente de limbaj: alfabet/vocabular, elemente lexicale, construcții sintactice. Pragmatici de programare. Paradigme de programare.

b. Cunoştinţele de specialitate

Programare logică. Fundamente teoretice – logica predicatelor de ordin unu. Aspectele sintactice şi semantice. Aspecte teoretice aplicate în Prolog. Principiul rezoluției. Unificarea. Reprezentarea cunoştinţelor în Prolog. Strategii de rezoluție în Prolog. Controlul rezoluției în limbajul Prolog.

Programare funcțională. Sintaxa calculului lambda. Semantica operațională. Arbori cu sintaxă abstractă. Limbajul ChezScheme.

c. Competentele generale

Studenții trebuie să posede competențe esențiale în rezolvarea de probleme prin metodă, analiză şi logică, aplicând gestiunea corespunzătoare a timpului şi capacități de mobilizare.

d. Competentele de specialitate

Studenții trebuie să fie capabili să identifice şi să aplice pragmatica de programare utilă în rezolvarea unui tip de probleme; să înțeleagă şi să adapteze paradigmele de programare logică şi funcțională.

e. Abilitățile cognitive specifice

Necesare pentru acest curs sunt eficacitatea personală a studenților prin adaptarea la noi situații, capacitatea de gestionare de situații, pragmatismul şi rigurozitatea de care dau dovadă studenții în rezolvarea de aplicații specifice.

3. Concordanța cu obiectivele planului de învățământ/specializării

a. Contribuția rezultatelor invatarii disciplinei la formarea competentelor specializării

Studenții vor fi deține cunoştinţe şi abilități specifice limbajelor, mediilor şi tehnicilor de programare logică şi funcțională.

Cunoştinţele pe care le vor deține le vor permite să analizeze şi să identifice paradigmele de programare logică şi funcțională, să modeleze şi să implementeze pragmaticele de programare specifice.

Studenții vor fi capabili să întocmească şi să gestioneze proiecte în domeniul informaticii aplicate folosind programarea logică sau funcională.

b. Cerințele disciplinare prealabile

Studenții trebuie să dețină cunoştinţe generale referitoare la caracteristicile programării procedurale, modul de lucru al limbajelor interpretate şi compilate, principiile programării, structuri de programare, tipuri şi structuri de date fundamentale sau definite de utilizator, logică matematică, algebră liniară şi analiză matematică.

4. Structura activității didactice

CURS ..... 42 ore
Seminar ..... 0 ore
Lucrări practice ..... 28 ore
Proiect ..... 0 ore

5. Prezentarea conținutului disciplinei

a. Curs

	Conținutul activității	*Nr. de ore*
1.	Obiectivele disciplinei. Clasificarea limbajelor de programare. Caracteristicile limbajelor de programare. Avantajele limbajelor de programare de nivel înalt. Scurtă prezentare a limbajelor procedurale/declarative şi a celor interpretate/compilate. Principiile programării. Structuri de programare (secvența, selecția simplă, selecția multiplă, repetitive).	3
2.	Tipuri şi structuri de date. Tipuri de date fundamentale (simple şi avansate, omogene – şiruri de date - vectori sau matrici, eterogene - înregistrările şi structurile). Tipuri de date definite de utilizator. Date statice şi dinamice. Exemple comparative.	3
3.	Taxonomia limbajelor de programare. Clasificarea limbajelor de programare după următoarele criterii: scopul propus şi aria de utilizare, scopul principal de aplicare, filozofia, stilul sau paradigma de programare. Elemente de limbaj: alfabet/vocabular, elemente lexicale, construcții sintactice.	3
4.	Concepte de programare. Sintaxa: definiție, gramatici, notații. Semantica: definiție, modalități de prezentare. Pragmatici de programare: iterația şi recursia, tablouri şi liste, exemple comparative. Paradigme de programare. Programare imperativă. Programare funcțională. Programare logică. Programare orientată obiect. Caracteristici.	3
5.	Programare logică. Fundamente teoretice – logica predicatelor de ordin unu. Aspectele sintactice (alfabetul limbajului predicatelor). Aspectele semantice: interpretarea formulelor bine formate; formule valide, inconsistente, echivalente; consecință logică;	3
6.	Aspectele semantice: clauze Horn; demonstrarea teoremelor prin reguli de inferență aplicând modus ponnens, modus tollens, specializarea universală.	3
7.	Aspecte teoretice aplicate în Prolog. Principiul rezoluției. Unificarea. Principiul rezoluției prin respingere.	3
8.	Cunoştințe în Prolog. Regulile. Faptele. Scopurile. Reprezentarea cunoştințelor în Prolog. Strategii de rezoluție în Prolog. Controlul rezoluției. Liste. Interpretare şi modele.	3
9.	Limbajul Prolog. Introducere. Predicate predefinite în Prolog. Fapte şi reguli - aplicații.	3
10.	Programare funcțională. Sintaxa calculului lambda.	3
11.	Semantica operațională. Arbori cu sintaxă abstractă.	3
12.	Limbajul ChezScheme. Introducere. Elementele de bază ale limbajului. Funcții predefinite	3
13.	Lucrul cu liste.	3
14.	Funții definite de utilizator. Structuri de control – utilizare, caracteristici.	3

b. Seminar

Conținutul activității	*Nr. de ore*

c. Lucrări practice

Conținutul activității	*Nr. de ore*
Exemple de aplicații ale limbajelor procedurale/declarative şi a celor interpretate/compilate. Implementarea principiilor programării şi a structurilor de programare.	2
Exemple comparative de lucru cu tipuri de date fundamentale (simple şi avansate, omogene – şiruri de date - vectori sau matrici, eterogene - înregistrările şi structurile) şi tipuri de date definite de utilizator.	2
Programare logică. Aplicații ale aspectelor semantice ale logicii predicatelor de ordin unu: interpretarea formulelor bine formate; formule valide, inconsistente, echivalente; consecință logică; clauze Horn; demonstrarea teoremelor prin reguli de inferență aplicând modus ponnens, modus tollens, specializarea universală;	2

Predicate predefinite în Prolog: verificarea tipului de date; funcţii de lucru cu variabile şi cu termenii; funcţii matematice; funcţii de control; operaţii cu clauze;	2
Cunoştinţe în Prolog. Regulile. Faptele. Scopurile. Reprezentarea cunoştinţelor în Prolog.	2
Aplicaţii de lucru cu formule logice. Aplicaţii de lucru cu arbori şi relaţii.	2
Strategii de rezoluţie în Prolog. Controlul rezoluţiei.	2
Lucrul cu liste în programarea logică.	2
Programare funcţională – introducere. Funcţii predefinite. Operatori.	2
Lucrul cu liste în programarea funcţională.	4
Funcţii definite de utilizator. Notaţia lambda.	2
Aplicaţii de utilizare a recursiei.	2
Structuri de control – utilizare, caracteristici.	2

d. Proiect

Conţinutul activităţii	*Nr. de ore*

6. Învăţare

a. Forme de învăţare/predare

Ca forme de predare se folosesc expunerea materialului de curs, care cuprinde aspecte teoretice şi exemplificarea acestora în aplicaţii practice la orele de laborator. Pentru o mai bună înţelegere a pragmaticilor de programare se realizează exemple comparative. La aplicaţiile practice se prezintă o serie de tipuri de probleme rezolvate, iar studenţii sunt îndrumaţi să folosească similaritatea pentru rezolvarea unor probleme asemănătoare.

b. Resurse educaţionale

Resursele on-line ale suportului de curs şi explicaţii ale aplicaţiilor de laborator prezentate ca model şi ale celor propuse spre rezolvare sunt prezente pe site-ul catedrei, la care au acces studenţii.

c. Bibliografie disponibila

Ţîndăreanu Nicolae, 1994, „Introducere în programarea logică. Limbajul Prolog", Editura Intarf.
Metadikes G.;. Nerode A, 1998, "Principii de logică şi programare logică", (trad. de A. Florea), Editura Tehnică
Trausanu-Matu St., 2004, "Programare în Lisp. Inteligenţă artificială şi web sematic", Editura Polirom.
Streinu I., 1986, "Lisp-Limbajul de programare al inteligenţei artificiale", Editura Ştiinţă.
Muscalagiu C., 1996, "Introducere în programarea logică şi limbajele de programare logică", Editura Univ. A.I.Cuza, Iaşi.

d. Alte resurse

Laboratoare cu dotări în calculatoare pe care sunt instalate Prolog şi ChezScheme.

7. Studiu individual: 80 ore

8. Evaluare

a. Forme de evaluare

A Pe perioada studiului studenţii sunt evaluaţi diagnostic formativ pentru monitorizarea progresului, a rezultatelor şi a nivelului de cunoaştere la care au ajuns. Se asigură astfel urmărirea cunoştinţelor dobândite de studenţi şi eventualele modalităţi de remediere necesare.

B La sfârşitul semestrului se realizează o evaluare sumativă prin examinare scrisă pentru verificarea cunoştinţelor teoretice ale programării logice şi funcţionale şi elaborarea practică şi susţinerea orală a unor aplicaţii de programare logică şi funcţională.

b. Principii de notare

Notarea la examen constă din 50% la proba scrisă şi 50% pentru elaborarea şi susţinerea aplicaţiilor practice.

c. Informarea studenţilor cu privire la evaluarea asociata disciplinei

Informarea studenților cu privire la modalitatea de desfășurare a predării disciplinei și modalitatea de examinare și notare se realizează la primul curs. De asemenea, studenții sunt atenționați să depună un efort susținut asupra aspectelor aplicative ale programării logice și funcționale.

9. Responsabil de curs

Nume : Ș.l. Cocu Adina
Date de contact : adina.cocu@ugal.ro, Universitatea „Dunărea de Jos", Galați, str. Domnească, Corp G, cam. G309.
Programul de contact se va desfășura conform programului afișat la avizierul catedrei.

Responsabil de curs,

Sef Departament / Catedra,