

FISA DISCIPLINEI
PROGRAMAREA CALCULATOARELOR
UG-FSC-C COBF103

Număr credite: 5

1. Obiectivele disciplinei

Studierea principiilor programării calculatoarelor prin intermediul limbajelor de programare C și C++. Însușirea gândirii algoritmice. Însușirea de către studenți a principiilor programării structurate. Formarea deprinderilor de programare și a capacității de elaborare de programe pentru rezolvarea unor probleme practice. Dezvoltarea și înțelegerea unor metode particulare de proiectare. Însușirea unui stil de programare profesionist. Însușirea de către studenți a principiilor programării structurate. Modelarea și construirea unor algoritmi clasici de rezolvare a unor probleme ingineresti. Modelarea și construirea unor algoritmi specifici de asigurare a fiabilității și portabilității programelor C. Înțelegerea și dezvoltarea unor metode particulare de proiectare. Formarea capacității de dezvoltare a aplicațiilor scrise în limbajul C pentru rezolvarea unor probleme tehnice.

2. Rezultatele învățării (se exprimă în obiective măsurabile ce fac subiectul evaluării)

a. Cunoștințele generale

La sfârșitul cursului, studenții trebuie să stăpânească fundamentele teoretice ale programării într-un limbaj de programare structurată (noțiunile de algoritm, dată, tip de dată, constantă, variabilă, structuri de control, modularizarea aplicațiilor complexe, programare structurată) și să fie capabil să elaboreze programe în limbajul C, pentru rezolvarea unor probleme practice.

b. Cunoștințele de specialitate

Studentul trebuie să își însușească gândirea algoritmică, fiind capabil să conceapă și să reprezinte un algoritm pentru o aplicație mică/medie.

Studentul trebuie să stăpânească noțiunile teoretice fundamentale pe care le va întâlni în orice limbaj de programare.

Studentul trebuie să dețină deprinderi de programare, fiind capabil să implementeze în limbajul C un algoritm pentru o problemă dată.

Studentul trebuie să cunoască algoritmi discutați la curs (Euclid, algoritmi de sortare: Bubble Sort, Quick Sort, etc.).

Studentul trebuie să își însușească un stil de programare profesionist, fiind capabil să modularizeze o aplicație (utilizând funcții, alocare dinamică de memorie).

c. Competențele generale

Nu este cazul

d. Competențele de specialitate

Competențe de dezvoltare: noțiuni de limbaje, noțiuni de algoritmi, noțiuni despre metode, norme și instrumente, noțiuni despre tehnici de dezvoltare a aplicațiilor informatice.

e. Abilitățile cognitive specifice

Nu este cazul

3. Concordanța cu obiectivele planului de învățământ/specializării

a. Contribuția rezultatelor învățării disciplinei la formarea competențelor specializării

Deoarece la această disciplină studenții dobândesc elementele teoretice fundamentale ale programării și deprinderea de a elabora programe, pe această disciplină se bazează toate disciplinele de informatică de la această specializare.

În cadrul acestei discipline, viitorii specialiști își vor forma vocabularul informatic, vor dobândi conceptele de bază și modul de gândire adaptat programării pe PC și UNIX workstation. Vor obține cunoștințele necesare programatorilor în limbajul C, în scopul rezolvării problemelor practice.

Cumulul de informații (hardware, algoritmice, software) dobândite reprezintă punctul de plecare (bază) la viitoarele discipline informatice (limbaje de programare orientate pe obiecte, structuri de date și tehnici de programare, ingineria sistemelor de programe, analiză și concepția sistemelor informatice, structuri și baze de date, etc.).

- b. Cerințele disciplinare prealabile
Nu este cazul

4. Structura activității didactice

CURS	28 ore
Seminar	14 ore
Lucrări practice	14 ore
Proiect	0 ore

5. Prezentarea conținutului disciplinei

a. Curs

<i>Continutul activității</i>	<i>Nr. de ore</i>
<u>1. Noțiuni introductive:</u> Calculatorul, un sistem automat de prelucrare a datelor. Algoritmi. Limbaje de programare. Teoria rezolvării problemelor. Etapele rezolvării unei probleme pe calculator.	(4 ore)
<u>2: Date, operatori și expresii:</u> Programe în limbajele C/C++. Preprocesorul. Elemente de bază ale limbajului. Date în limbajele C/C++ (tipuri de date, constante, variabile). Conversii.	(4 ore)
<u>3: Implementarea structurilor de control:</u> Principiile programării structurate. Implementarea structurilor de control (secvențială, alternativă, repetitivă). Facilități de întrerupere a unei secvențe.	(4 ore)
<u>4: Tablouri:</u> Tablouri unidimensionale. Tablouri multidimensionale. Șiruri de caractere. Algoritmi de prelucrări numerice pe tablouri. Metode de sortare.	(3ore)
<u>5: Pointeri:</u> Variabile pointer. Operații cu pointeri. Pointeri și tablouri. Tablouri de pointeri. Pointeri la pointeri. Modificatorul const în declararea pointerilor.	(3 ore)
<u>6: Funcții:</u> Structura unei funcții. Transferul parametrilor unei funcții (prin valoare, pointeri și referință). Funcții cu parametri impliciți. Funcții cu număr variabil de parametri. Funcții predefinite. Supraincercarea funcțiilor. Clase de memorare. Moduri de alocare a memoriei. Funcții recursive. Pointeri către funcții.	(6 ore)
<u>7: Tipuri de date definite de utilizator:</u> Structuri. Campuri de biți. Declarații de tip. Uniuni. Enumerari.	(2 ore)
<u>8: Fișiere:</u> Caracteristici generale. Deschiderea și închiderea unui fișier. Prelucrarea fișierelor text. Intrări-iesiri binare. Funcții utilitare pentru lucrul cu fișierele. Alte operații cu fișiere.	(2 ore)

b. Seminar

<i>Continutul activității</i>	<i>Nr. de ore</i>
Reprezentări ale datelor în binar.	(2 ore)
Conceperea și descrierea unor algoritmi.	(2 ore)
Date, operatori, expresii	(2 ore)
Implementarea în limbajul C a unor algoritmi	(2 ore)
Implementarea unor algoritmi utilizați pentru tablouri numerice (prelucrări numerice, calcul vectorial și matriceal)	(2 ore)
Modularizarea programelor; utilizarea funcțiilor	(2 ore)
Aplicații complexe	(2 ore)

c. Lucrări practice

<i>Continutul activității</i>	<i>Nr. de ore</i>
Prezentarea calculatorului și a mediului de programare în care se va lucra. Structura unui program C și exemple simple.	(2 ore)

Aplicatii simple cu tipuri de date predefinite, operatori si expresii.	(2 ore)
Implementarea structurilor de control. Implementarea unor algoritmi pentru diverse probleme.	(2 ore)
Tablouri. Probleme cu vectori, matrici, siruri de caractere si prelucrari ale elementelor acestora.	(2 ore)
Algoritmi de procesare a sirurilor de caractere interpretate ca tablouri de caractere.	
Pointeri. Aplicatii in care se utilizeaza functiile predefinite si implementarea de functii utilizator; tipuri de date definite de utilizator	(4 ore)
Fisiere. Aplicatii cu fisiere.	(2 ore)

d. Proiect

Continutul activitatii

Nr. de ore

6. Invatare

a. Forme de invatare/predare

Predarea cursului se face sub formă de prelegere, combinata cu dialogul. Cunoștințele se completează la laborator si la seminar, când are loc participarea directă a studenților la dezbaterile lucrărilor, rezolvarea temelor de laborator.

descoperirea dirijata, problematizarea, studiul de caz, conversatia euristica, demonstratia, dialogul, observatia.

Metodele pedagogice folosite: Conversatia euristica, expunerea, problematizarea, invatarea prin rezolvarea de probleme, studiul de caz, dialogul, inductia.

b. Resurse educationale

1. Stefanescu D., Curs C/C++, I si II, www.ariadne.ugal.ro
2. Stefanescu D., Indrumar de laborator la disciplina limbaje de programare, I si II, www.ariadne.ugal.ro

c. Bibliografie disponibila

1. Brookshear J.G., Introducere in informatica, Ed. Teora, Bucuresti, 1998
2. Cristea V., s.a. Limbajul C standard, Ed. Teora, 1992.
3. Jamsa K, Klander L, Totul despre C si C++ - Manualul fundamental de programare, Ed. Teora, Bucuresti, 2005
4. Kernighan B., Ritchie D., Limbajul C, Ed. Teora, Bucuresti, 2005
5. Negrescu L., Limbajul C si C++ pentru incepatori, vol. I si II, Ed. Teora, Bucuresti, 2003
6. Stefanescu D., Programarea in limbajele C/C++, MATRIXRom, Bucuresti, 2002
7. Stefanescu D., Segal C., Initiere in limbajele C/C++, Editura Fundatiei Universitare "Dunarea de Jos", Galati, 2000

d. Alte resurse

7. Studiu individual: 100 ore

8. Evaluare

a. Forme de evaluare

Evaluare formativa si cu caracter de diagnosticare, realizata in cadrul activitatii de laborator, prin: teste teoretice periodice si discutarea rezultatelor, urmarirea modului in care studentii rezolva problemele propuse cu scopul de a depista si corecta greselile acestora (nota N11).

Evaluare sumativa, realizata prin verificarea practica finala de la laborator (N12) si examenul scris, final (nota N2).

b. Principii de notare

- Notarea activitatii din timpul anului: N1
 - Verificarea sistematica a programelor elaborate de studenti in timpul lucrarilor practice: N11
 - Verificarea finala lab.: N12
 - $N1=0.3*N11+0.7*N12$

- Lucrare scrisa la examen: N2
 - Teorie – punctaj maxim 10 p, punctaj min promovare 5 p, nota N21
 - Probleme – punctaj maxim 10 p, punctaj min promovare 5 p, nota N22
 - $N2=0.5*N21+0.5*N22$

➤ Nota finala: $NF=0.25*N1 + 0.75*N2$

OBS:

Prezentarea la examen este conditionata de notarea activitatii (cu min. 5) din timpul anului.

Examenul este scris (2.5 ore)

La proba scrisa, studentul trebuie sa acumuleze min 5 puncte atat la teorie, cat si la probleme.

c. Informarea studentilor cu privire la evaluarea asociata disciplinei

La primul curs, se prezinta studentilor modul in care vor fi evaluati si notati la disciplina respectiva.

9. Responsabil de curs

Nume : s.l.ing. STEFANESCU DIANA

Date de contact : Diana.Stefanescu@ugal.ro

Facultatea de Stiinta Calculatoarelor, Str. Domneasca 111, Corpul G, et. III, G 310, 800201 Galati, Tel./fax:
+40 236 460182

Program de contact: luni 7:30-14, marti 8-19, miercuri, joi 8-14

Responsabil de curs,

Stefanescu

Sef Departament / Catedra,

