

**MINISTERUL EDUCAȚIEI AL REPUBLICII MOLDOVA
UNIVERSITATEA DE STAT „ALECU RUSSO” DIN BĂLȚI
FACULTATEA DE ȘTIINȚE REALE, ECONOMICE ȘI ALE MEDIULUI
CATEDRA DE MATEMATICĂ ȘI INFORMATICĂ**

CURRICULUM

pentru unitatea de curs

„LIMBAJE DE PROGRAMARE STRUCTURATĂ”

pentru specialitățile Informatică, Informatică (științe exacte)

Ciclul I, studii superioare de licență, învățământ cu frecvență

Domeniul de formare: Științe ale educației

pentru specialitățile:

141.02. Informatica,

141.01. Matematica și 141.02. Informatica

141.03. Fizica și 141.02. Informatica

secția de învățământ cu frecvență la zi

Titularii disciplinei:

lector superior universitar Vitalie Țicău,

doctor, lector universitar Diana Moglan

BĂLȚI, 2016

Curriculumul a fost discutat la ședința Catedrei de matematică și informatică

Procesul verbal nr. 1 din 29.08.2016

Șeful catedrei dr. conf. univ. E. Plohotniuc _____

Curriculumul a fost aprobat la ședința Consiliului Facultății de Științe Reale, Economice și ale Mediului

Procesul verbal nr. 5 din 20.10.2016

Decanul facultății, dr. hab., prof. univ. P. Topală _____

Informații de identificare a disciplinei

Facultatea: Științe Reale, Economice și ale Mediului

Catedra: Matematică și informatică

Domeniul general de studiu: 14 Științe ale Educației

Domeniul de formare profesională la ciclul I: 141 Educație și formarea profesorilor, ciclul I, licență

Denumirea specialității: Informatica, Matematica și Informatica, Fizica și Informatica

Administrarea unității de curs

Spec.	Codul unității de curs	Credit e ECTS	Total ore	Repartizarea orelor				Forma de evaluare	Limba de predare
				Prelegeri	Seminare	Laboratoare	Lucrul individual		
IP	S.02.O.211	6	180	30	–	60	90	Examen	Rom
MI	S2.02.O.211	5	150	30	–	60	60	Examen	Rom
FI	S2.02.O.211		150	30	–	60	60	Examen	Rom

Statutul: Unitate de curs obligatorie de specializare

Localizarea sălilor: curs – aulele 505, 101, 224, laborator – aulele 140, 141, 146, 150

Informații referitoare la cadrele didactice

Titularii cursului – **Țicău Vitalie**, lector superior universitar. Absolvent a Universității de Stat din Moldova, specialitatea „Matematica aplicată”. A finalizat studiile de doctorat la specialitatea „Matematica de calcul”. A realizat publicații aplicative și metodice cu tematica: „Structuri discrete”, „Limbaje de programe orientate obiect”, „Analiza numerică”, „Prelucrarea informației grafice”, „Rezolvări de probleme din domeniul informaticii”. Formator permanent din anul 2003 în cadrul cursurilor de formare continuă a profesorilor de informatică.

Sediul – aula 145. Tel. 0 231 52 488.

E-mail: VitalieSTicau@gmail.com

Orele de consultații – marți, joi: 14.30-16.00. Consultațiile se oferă atât în regim „față-în-față”, cât și prin utilizarea poștei electronice.

Moglan Diana, dr. în științe pedagogice, lector universitar, absolventa Universității de Stat „A. Russo” din Bălți, specialitatea „Matematica și informatica”. A efectuat studiile de master la specialitatea „Informatica”. A susținut teza de doctor în pedagogie la specialitatea „Teoria și metodologia instruirii (științe exacte și ale mediului)”.

E-mail: di_2008@mail.ru

Skype: mogdiana

Orele de consultații – miercuri: 15.00-16.30. Consultațiile se oferă în regimul „față-în-față”, prin utilizarea poștei electronice și prin Skype.

Integrarea cursului în programul de studii (planul de învățământ)

Programele de studii „Informatica” „Matematica și Informatica” „Fizica și Informatica” sunt programe de studii din domeniul 141, Științe ale Educației și absolventul acestei specialități este profesor de informatică, matematică și informatică, fizică și informatică. Astfel, cursul de programare structurată reprezintă un curs de specializare, care vine cu scopul dezvoltării competenței de programare a studenților, precum și a competenței de predare a disciplinei „Informatica” în școală pentru specialitățile pedagogice.

Cursul respectiv dezvoltă formarea competențelor profesionale ale viitorului profesor de matematică sau inginer-programator, analizând specificul formării deprinderilor informatice atât în aspect general, cât și particular pe domeniu.

Cursul "Limbaje de programare structurată" este o continuare a cursului fundamental „Bazele programării” și contribuie la dezvoltarea competenței de programare a studenților aplicând diverse limbaje de programare, în special limbajul C/C++. Acest curs este intermediar și are o continuitate în cursurile „Programare orientată pe obiect I”, „Programare orientată pe obiect II”, „Analiza numerică”, etc.

Prin conținutul său și activitățile de învățare a studenților, unitatea de curs „Limbaje de programe structurate” contribuie la dezvoltarea mai multor competențe generice, necesare profesorului de informatică:

- capacitatea de analiză și sinteză;
- deprinderi de comunicare în limba maternă;
- capacitatea de a lucra în echipă;
- capacitatea de a aplica cunoștințele în practică;
- capacitatea de a genera idei noi;
- capacitatea de a lucra independent.

Competențe prealabile

1. Utilizarea eficientă a resurselor sistemelor de calcul, de operare și ale Internetului.
2. Posedarea la nivel teoretic și aplicativ a informaticii preuniversitare în limitele standardelor de studiu eficient al învățării (sursa electronică www.edu.md);
3. Posedarea deprinderilor de elaborare și analiză a algoritmilor de rezolvare a problemelor practice în baza precizării problemei;
4. Posedarea deprinderilor de lucru cu MS OFFICE: elaborare de documente WORD, prezentări Power Point etc.

Competențe dezvoltate în cadrul cursului

În cadrul studierii unității de curs studenții își vor dezvolta următoarele competențe (CP – competențe profesionale; CT – competențe transversale):

Competențe profesionale:

CP2.1. Identificarea tipurilor de date și a structurii modelelor informatice și didactice pentru descrierea unor fenomene și procese reale.

CP3.3. Aplicarea de principii și metode de bază din informatică și științe ale educației pentru proiectarea și elaborarea unor algoritmi specifici domeniului profesional.

CP4.4. Utilizarea adecvată de criterii și metode standard de evaluare, pentru a aprecia calitatea, avantajele și limitele proceselor de programare, dezvoltare și mentenanță ale aplicațiilor informatice în limbaje de nivel înalt.

Competențe transversale:

CT1. Aplicarea regulilor de muncă riguroasă și eficientă, manifestarea unei atitudini responsabile față de domeniul științific și didactic, pentru valorificarea optimă și creativă a propriului potențial în situații specifice, cu respectarea principiilor și a normelor de etică profesională.

CT2. Desfășurarea eficientă și eficace a activităților organizate în echipă.

Finalitățile cursului

La finalizarea studierii unității de curs studentul va fi capabil:

- să proiecteze algoritmul și etapele de elaborare și rezolvare a problemei.
- să rezolve probleme concrete, bazate pe aplicarea tipurilor de date omogene și mixte.
- să rezolve probleme concrete evidențiind aplicarea structurilor dinamice și statice de date.
- să rezolve probleme concrete prin aplicarea algoritmilor clasici de programare;
- să lămurească pas cu pas îndeplinirea programului în baza exemplurilor concrete;
- să analizeze corectitudinea aplicării uneia sau alteia structuri de control în program.

Structura unității de curs

Nr. D/o	Denumirea și conținutul scurt al temei	Curs, ore
Unitatea de conținut 1: Structura unui program. Instrucțiuni. Tipuri de date simple, omogene		
1.	Generalități privind limbajele de programare structurată. Medii de programare. Sintaxa și alfabetul limbajului. Cuvintele cheie. Blocuri de instrucțiuni.	1
2	Tipuri de date. Constante și variabile. Declararea variabilelor. Adresele și inițializarea variabilelor.	1
3	Funcțiile de citire și de scriere a datelor în mod standard.	1
4	Operatori: aritmetici, de atribuire, de incrementare și decrementare, relaționali, logici, la nivel de bit, condițional, de rezoluție.	1
5	Instrucțiuni de decizie: <i>if</i> , <i>if-else</i> , <i>switch</i> .	1
6	Instrucțiuni de ciclare: <i>for</i> , <i>while</i> , <i>do-while</i> . Clasificarea ciclurilor. Instrucțiunile <i>break</i> și <i>continue</i> , <i>goto</i> , <i>exit</i> .	2
7	Definirea și declararea funcțiilor. Apelul funcțiilor. Parametri formali și actuali. Transmiterea parametrilor prin: valoare, adresă, referință. Variabile	2

	globale și locale. Variabile automate, statice, externe. Durata de viață a variabilelor.	
8	Directive de preprocesare: #define, #include, #if, #ifdef, #ifndef, #endif, #else.	1
9	Pointeri. Declararea și utilizarea pointerilor. Utilizarea pointerilor ca parametri ai funcțiilor. Mecanismul de indirectare.	1
10	Tabele. Tabele unidimensionale. Inițializarea tabelului. Accesarea elementelor unui vector. Utilizarea tabelelor ca pointeri. Algoritm de căutare în vector. Determinarea valorii minime și maxime. Algoritm de formare a unui vector din alți vectori. Algoritm de deplasare a elementelor unui vector. Algoritmi de rotire a elementelor unui vector. Algoritmi de sortare a vectorilor.	2
11	Tabele bidimensionale. Accesul la elementele tabelului. Prelucrarea tabelelor pe rânduri sau coloane. Algoritmi de formare a tabelelor. Determinarea elementelor minime și maxime. Includerea unui rând / unei coloane în tabel. Excluderea unui rând sau a unei coloane din tabel. Generarea matricelor.	2
Unitatea de conținut 2: Șiruri de caractere. Tipuri de date derivate. Apeluri recursive		
12	Șiruri de caractere. Utilizarea șirurilor de caractere ca pointeri. Funcții standard de prelucrare a șirurilor de caractere.	2
13	Structuri și uniuni. Accesarea elementelor structurii. Structuri înălțuite. Inițializarea structurilor. Redenumirea tipurilor de date. Pointeri la structuri. Câmpuri. Tipul enumerare.	2
14	Fișiere. Modul standard de intrare-ieșire. Clasificarea fișierelor. Deschiderea și închiderea unui fișier. Modul de acces la deschiderea fișierelor. Scrierea și citirea fișierelor de tipul: caracter, șir de caracter, formatat, înregistrare. Accesarea directă și secvențială a articolelor unui fișier. Fișiere standard.	2
15	Recurсія. Recursie directă și indirectă.	1
16	Supraîncărcarea funcțiilor. Funcții inline. Funcții cu parametri implicați.	1
17	Structuri dinamice de date. Stiva. Coadă. Lista liniară. Liste dublu înălțuite și liste dublu înălțuite cu santinele. Implementarea stivelor, cozilor, listelor. Operațiile posibile cu o structurile dinamice de date.	2
18	Elaborarea programelor complexe. Fișiere proiect. Etapele de realizare a unui program complex. Modele de memorie. Tipurile de pointeri.	2
Unitatea de conținut 3: Elemente de programare orientată pe obiecte		
19	Elemente fundamentale ale POO. Clase și obiecte. Modificatorii de protecție: private, protected și public. Mesaje și metode. Constructori. Destructorii. Funcții prietene. Operatori supradefiniți și operatori prieteni.	2
20	Moștenirea. Polimorfismul. Ierarhii de clase. Funcții virtuale. Supradefinirea funcțiilor membre și a operatorilor. Clase prietene. Moștenirea multiplă. Clase abstracte.	1
Total		30

Tematica și repartizarea orientativă a orelor la laborator

Orele de laborator sunt organizate în formă de realizare a 7 proiecte de program și o lecție de inițiere. În următorul tabel este prezentată tematica proiectelor de program și repartizarea orelor.

Nr. D/o	Tema	Nr. de ore
	Mediul de programare. Înscrierea programelor. Compilarea. Link - editarea. Lansarea în execuție. Instrucțiuni de intrare – ieșire. Trasarea pe pași a programului. Utilizare Help.	2
1	Proiect de programe Nr 1. Organizarea operațiilor la nivel de bit. Algoritmi clasici. Organizarea ciclurilor în program.	8
2	Proiect de programe nr. 2. Organizarea ciclurilor în program. Gestiunea tabelelor de numere. Pointeri. Funcții	8
3	Proiect de programe nr. 3. Gestiunea tabelelor bidimensionale de numere.	8
4	Proiect de programe nr. 4. Șiruri de caractere. Fișiere text.	8
5	Proiect de programe nr. 5. Structuri. Fișiere binare.	8
6	Proiect de programe nr. 6. Structuri dinamice.	8
7	Proiect de programe nr. 7. Fișiere proiect. Clase.	8
	Evaluare	2
Total		60

Fiecare student îndeplinește un set de acțiuni conform setului specificat de proiecte de program, unde sunt indicate: scopul lucrării; formularea problemei; datele inițiale; indicații metodice și recomandări; cerințele față de program; variantele însărcinărilor.

Activități de lucru individual

Sarcini pentru activitate individuală sunt propuse la finele fiecărei teme din cursul electronic și se precizează la sfârșitul fiecărei prelegeri. În special sunt prezente sarcini de implementare a materialului teoretic cu aplicații în practică. Exemple:

1. De determinat nota maximă (minimă) a studenților dintre notele la o disciplină, înscrise într-un tabel unidimensional.
2. În prelungire, de determinat nota maximă (minimă) a studenților dintre la mai multe disciplină, înscrise într-un tabel bidimensional. De legat cu un tabel unidimensional cu numele studenților.
3. În continuare, datele se citesc din fișier textual, apoi binar.
4. Și, în final, datele se protejează, folosind clasele și conceptele programării orientate pe obiect.

Consultațiile pentru lucru individual sunt expuse în orarul catedrei.

La finele cursului, la dorință, unii studenți pot să elaboreze un program complex, care se susține public în fața titularului de curs, a profesorului de la orele de laborator și a grupei academice. Condițiile inițiale pentru programul complex sunt la fel expuse pe platforma de învățare MOODLE. La preferința studentului și cu acordul profesorului condițiile programului pot fi modificate pentru a fi mai captivante. Se recomandă o prezentare PowerPoint pentru inițierea în tipurile de date utilizate și algoritmul utilizat.

Evaluable

Evaluable curentă:

Evaluable curentă se realizează în cadrul orelor de curs și de laborator.

La prelegeri se realizează două lucrări de control. De asemenea studenții pot fi evaluați la orele de prelegeri prin rezolvări de probleme, argumentări a elaborării algoritmului de calcul.

Prima lucrare de control este evaluată pe platforma MOODLE la calculator în formă de test pe unitatea de conținut nr. 1.

A doua lucrare de control este petrecută în scris în formă de test cu 4 itemi de rezolvare și 2 sarcini practice de definire a unei funcții pe unitatea de conținut nr. 2 și ultimele teme din unitatea de conținut nr. 1.

La laborator se evaluează cele 7 proiecte de program și o lecție de inițiere. Fiecărui student îi este prezentat modul de evaluare a proiectelor.

Se determină mediile notelor de la prelegeri și de la laborator.

Nota la evaluare curentă (NEC) se determină ca media notelor de la prelegeri și de la laborator.

Evaluable finală:

Evaluable finală se realizează în formă de examen scris – test cu itemi similari de la lucrările de control de la evaluare curentă. În total sunt 7 itemi de rezolvare și 3 sarcini practice de definire a unei funcții pe unitățile de conținut nr. 1 și nr. 2.

Durata desfășurării examenului – 2 ore astronomice. Studentul la examen este apreciat cu o notă (NE) conform baremului.

Nota finală (NF) se calculează după formula:

$$NF = 0.6 \times NEC + 0.4 \times NE.$$

Chestionarul pentru examen conține doar temele indicate în conținuturi.

Resurse informaționale ale cursului:

Obligatorie:

1. NEGRESCU, L. *Limbele C și C++ pentru începători. Volumul 1. Limbajul C. Volumul 2. Limbajul C++*. Cluj, 1996.
2. BARKAKATI, N. *Borland C++*. Ghidul programatorului. București: Teora, 1997.
3. BALTA, I. *Programarea în limbajele C și C++*. București, Teora, 1996.
4. BOTEZ, C. GOSPODARU, M. *Limbe de programare. Introducere în limbajul C*. Iași, 1998.
5. NAMIR, C. Shammass. *Curs rapid de Borland C++ 4*. București: Teora, 1996 (trad.engl.).
6. СЕДЖВИК, С. *Язык программирования Си++*. М.: Мир, 2001.
7. СТРАУСТРУП, Б. *Язык программирования C++*, 3-е изд. СПб.: Невский Диалект, 1999.
8. ПОЛ, А. *Объектно-ориентированное программирование на C++*. СПб.: Питер, 2001.

Suplimentară:

1. RÎNGACI, V. D.; LATUL, G. V.; SOCOL V. V. *Programare în limbajul Turbo C. (în 3 părți)*. Chișinău, 1994.
2. BOTEZ, C.; NEAGA, C. *Limbajul de programare C. Exemple*. Iași, 1997.

3. БАДД, Т. *Объектно-ориентированное программирование в действии*. СПб.: Питер, 1997.

Exemplu de test pentru evaluare curentă

Catedra de matematică și informatică

Aprob:

Șeful catedrei _____

Proba Nr. 2 de evaluare curentă la unitatea de curs “*Limbaie de programare structurată*”

În itemii 1-4: indicați, ce va fi afișat pe ecran și lămuriți rezultatul afișat (2p). Identificați și descrieți algoritmul definit în funcția <i>func</i> (2p). Indicați datele de intrare și de ieșire (2p).	
1. Tema: Tabele unidimensionale (6 p) <pre>int func(int, int []); int main() { int i, n = 10; int x[] = {2, 10, -4, 3, 15, 23, 25, 5, 6, 8}; cout << func(n, x); return 1; } int func(int n, int a[]) { int i; int aux = a[0]; for(i=1; i<n; i++) if(a[i] > aux) aux = a[i]; return aux; }</pre> a) 10 b) 25 c) -4 d) 2	2. Tema: Șiruri de caractere (6 p) <pre>int func(char []); int main() { char fraza[81] = "In anii 2000-2015 C++ ramine inca un limbaj popular!"; cout << func(fraza); return 1; } int func(char sir[]) { int n = 0; char *cuvint; if((cuvint = strtok(sir, " -!+")) == NULL) return 0; n = 1; while((cuvint = strtok(NULL, " -!+")) != NULL) n++; return n; }</pre> a) 9 b) 10 c) 11 d) 12
3. Tema: Structuri (6 p) <pre>struct Punct { double x; double y; }; int func(int, Punct []); int main() { int n = 10; Punct p[10] = {{1.0, 2.5}, {1.0, 2.0}, {1.0, -2.5}, {-1.0, 2.5}, {0.0, 2.5}, {3.0, -5.0}, {1.0, 0.0}, {0.0, 0.0}, {1.0, 10.0}, {-1.0, -2.5}}; cout << func(n, p); return 1; } int func(int n, Punct p[]) { int i, k = 0; for(i=0; i<n; i++) if((p[i].x >= 0) && (p[i].y >= 0))</pre>	4. Tema: Fișiere (6 p) Conținutul fișierului <i>date.in</i> <pre>1 Ion 2200 2 Gheorge 1800 3 Elena 3500 4 Iulian 5200 5 Cristian 2500 6 Sandu 1500 7 Ionela 6000 8 Ala 2350 9 Tanea 1800 10 Alina 3300 char *func(); struct Angajat { int id; char nume[11]; double salariu; }; int main() { cout << func(); return 1; } char *func() { FILE *f; char sir[11]; Angajat a; double aux = 1e+38; f = fopen("date.in", "rb"); while(fread(&a, sizeof(Angajat), 1, f) > 0) if(a.salariu < aux) {</pre>

<pre>k++; return k; }</pre> a) 5 b) 6 c) 7 d) 8	<pre>aux = a.salariu; strcpy(sir, a.nume); }</pre> <pre>fclose(f); return sir; }</pre> a) Ion b) Gheorge c) Sandu d) Ionela
--	--

5. Elaborarea programului (10 p) Se dă un tabel, format din $N \leq 100$ elemente. Scrieți o funcție, în care se inserează între fiecare două elemente media lor. Funcția întoarce tabelul rezultat. De apelat această funcție în funcția *main*. // Задан массив из $N \leq 100$ элементов. Напишите функцию для вставки между любыми двумя элементами их среднего значения. Функция возвращает полученный массив. Вызвать эту функцию в функцию *main*.

6. Elaborarea programului (10 p) Se consideră o matrice $A(N \times N)$, ($N \leq 20$). Scrieți o funcție, care modifică matricea astfel că toate elementele acesteia se măresc cu valoarea elementului situat pe diagonala principală și în coloana în care se află elementul modificat. Funcția întoarce noua matrice. // Задана матрица $A(N \times N)$, ($N \leq 20$). Напишите функцию, в которой прибавляется к каждому элементу матрицы значение элемента находящегося на главной диагонали в соответствующем столбце. Функция возвращает новую матрицу.

Punctaj	0-1	2-4	5-10	11-15	16-19	20-23	24-30	31-36	37-41	42-44
Nota	1	2	3	4	5	6	7	8	9	10

2016

Titularul cursului: _____

Exemplu de test pentru evaluare finală

Aprob:

Șeful catedrei _____

Testul Nr 1

Pentru examen la disciplina “Limbaje de programare structurată”

În itemii 1-5: indicați, ce va fi afișat pe ecran și lămuriți rezultatul afișat (2p). Identificați și descrieți algoritmul definit în funcția *func* (2p). Indicați datele de intrare și de ieșire (2p).

1. Tema: Instrucțiuni de decizie (6 p) <pre>void func(int &, int &); int main() { int a = 12, b = 5; func(a, b); cout << a << b; return 1; } void func(int &a, int &b) { if(a >= 10) if(a == 15) ++b; else a = 2; b--; }</pre> a) 12 5 b) 2 5 c) 12 4 d) 2 4	4. Tema: Structuri (6 p) <pre>struct Student { int id; char nume[25]; int nota[4]; }; int func(int, Student []); int main() { int n = 10, k; Student grupa[10] = {{1, "Ana", {8,10,10,10}}, {2, "Ala", {10, 10,2,10}}, {3, "Ionela", {10,10,10,10}}, {4, "Ion", {5,5,5,5}}, {5,"Valeriu",{9,9,9,9}}, {6,"Igor",{6,6,6,6}}, {7,"Tanea",{8,7,8,7}},</pre>
--	---

2. Tema: Tabele unidimensionale (6 p) <pre>int func(int, int []); int main() { int i, n = 10; int x[] = {2, 10, -4, 3, 15, 23, 25, 5, 6, 8}; cout << func(n, x); return 1; } int func(int n, int a[]) { int i; int aux = a[0]; for(i=1; i<n; i++) if(a[i] > aux) aux = a[i]; return aux; }</pre> a) 10b) 25c) -4d) 2	<pre>{8,"Irina",{8,8,8,6}},{9,"Olga",{8,8,5,8}},{10,"Elena", {10,7,10,10}}; k = func(n, grupa); cout << k; return 1; } int func(int n, Student grupa[]) { int i, j, m, k = 0; float aux = 0; for(i=0; i<n; i++) { m = 0; aux = 0; for(j=0; j<4; j++) if(grupa[i].nota[j] >= 5) { m++; aux += grupa[i].nota[j]; } if(m == 4) aux /= 4; if(aux >= 7.5) k++; } return k; }</pre> a) 5b) 6c) 7d) 8												
3. Tema: Șiruri de caractere (6 p) <pre>int func(char []); int main() { char fraza[81] = "In anii 2000-2015 C++ ramine inca un limbaj popular!"; cout << func(fraza); return 1; } int func(char sir[]) { int n = 0; char *cuvint; if((cuvint = strtok(sir, " -!+") == NULL) return 0; n = 1; while((cuvint = strtok(sir, " -!+") != NULL) n++; return n; }</pre> a) 9 b) 10c) 11d) 12	5. Tema: Fișiere (6 p) Conținutul fișierului <i>date.in</i> <table><tr><td>1 Ion 2200</td><td>2 Gheorge 1800</td><td>3 Elena 3500</td></tr><tr><td>4 Iulian 5200</td><td>5 Cristian 2500</td><td>6 Sandu 1500</td></tr><tr><td>7 Ionela 6000</td><td>8 Ala 2350</td><td>9 Tanea 1800</td></tr><tr><td>10 Alina 3300</td><td></td><td></td></tr></table> <pre>char *func(); struct Angajat { int id; char nume[11]; double salariu; }; int main() { cout << func(); return 1; } char *func() { FILE *f; char sir[11]; Angajat a; double aux = 1e+38; f = fopen("date.in", "rb"); while(fread(&a, sizeof(Angajat), 1, f) > 0) if(a.salariu < aux) { aux = a.salariu; strcpy(sir, a.nume); } fclose(f); return sir; }</pre> a) Ionb) Gheorgec) Sandud) Ionela	1 Ion 2200	2 Gheorge 1800	3 Elena 3500	4 Iulian 5200	5 Cristian 2500	6 Sandu 1500	7 Ionela 6000	8 Ala 2350	9 Tanea 1800	10 Alina 3300		
1 Ion 2200	2 Gheorge 1800	3 Elena 3500											
4 Iulian 5200	5 Cristian 2500	6 Sandu 1500											
7 Ionela 6000	8 Ala 2350	9 Tanea 1800											
10 Alina 3300													
6. Elaborarea programului (10 p) De declarat și de definit o funcție, care întoarce numărul și suma numerelor prime din numerele întregi citite de la tastatură până la introducerea valorii 0. De apelat funcția în <i>main</i>. // Объявить и определить функцию, которая возвращает число и сумму простых чисел из введенных с клавиатуры до ввода 0. Вызвать функцию в <i>main</i>.													
7. Elaborarea programului (10 p) Se consideră o matrice $A(N \times N)$, ($N \leq 20$). Scrieți o funcție, care modifică matricea astfel că toate elementele acesteia se măresc cu valoarea elementului situat pe diagonala principală și în coloana în care se află elementul modificat. Funcția întoarce noua matrice. // Задана матрица $A(N \times N)$,													

($N \leq 20$). Напишите функцию, в которой прибавляется к каждому элементу матрицы значение элемента находящегося на главной диагонали в соответствующем столбце. Функция возвращает новую матрицу.

8. Algoritmi (10 p). Definiți și descrieți verbal (prin comentarii a secvențelor) algoritmul de verificare dacă numărul indicat este sau nu palindrom. // Определить и описать словесно алгоритм проверки, является ли указанное число палиндромом.

2016

Examinator: _____