Teste grila propuse pentru examenul de licenta 1999 la sectia de Informatica Economica, facultatea CSIE

1.

```
#include "iostream.h"
#include "stdarg.h"
class cls
{public: int a;  cls(int v):a(v){ }
         void g(int); };

class sbcls
{int y;
 public: sbcls(int v=2):y(v) { }
 void f(int,...); };

void cls::g(int n)
{sbcls s; s.f(1,this);     }

void sbcls::f(int t,...)
{va_list lv; va_start(lv,t);
  cls *h;
  if(t = =1) h = (cls *) va_arg(lv,void *);
         cout << h->a;  va_end(lv);
}

void main( )
{ cls t(7); t.g(1);  }
```

În contextul de mai sus, *this* reprezinta:

a) pointer la un obiect de tip cls;
b) obiect de tip cls;
c) pointer la un obiect de tip sbcls;
d) obiect de tip sbcls;
e) pointer la ferestra principala a aplicatiei.

2.

Fie declaratia :

```
#include <iostream.h>
class c1    { int a; } ;
class c2 : public c1
{public:
  int b;
  void scrie_a(){cout<<"a="<<a<< endl;}
} ;
void main()
{c2 ob; ob.scrie_a (); }
```

Selectati afirmatia corecta:

a) functia de acces nu are drept de scriere a unui membru privat;
b) programul afiseaza valoarea lui a;
c) functia de acces are doar acces *read-only* asupra unui membru privat;
d) derivarea publica este incorect realizata
e) prin derivare publica, accesul la membrii mosteniti devine public.

3.

Se da clasa :

```
class c  {  double a, b ;
public :
         friend c operator + (c &, double ) ;
         friend c operator + (double, c & ) ;
         } ;
```

În declaratia de mai sus apare:

a) o supraîncarcare prin functii independente;
b) o supraîncarcare prin functii membre în clasa c;
c) o supraîncarcare de constructori de clasa c;
d) o supraîncarcare de constructori de copiere ai clasei c;
e) declararea unor clase *friend*.

4.

```
#include <iostream.h>
class ex1 ;
class ex2
{int a;
 public: int b;
         friend class ex1 ;
protected: int c;
} ;
class ex1
{public: void f(ex2 &ob){cout<<ob.a<<
ob.b <<ob.c; }  };

void main()
{ ex1 ob1; ex2 ob2;    ob1.f(ob2);      }
```

Precizati care din afirmatiile de mai jos este cea corecta.
a) f() are acces la toti membrii clasei ex2;
b) f() are acces doar la variabila publica b, deoarece declaratia de friend este data în domeniul public;
c) f() are acces doar la variabilele publice si protected (varabilele b si c);
d) f() nu are acces la nici una din variabilele a,b,c.
e) functia f() este incorect apelata.

5.

```
#include <iostream.h>
struct persoana
{char nume[50];
  struct copil {char prenume[10]; int virsta;
} c[3];
} p1,*pp;
void main()
{pp=&p1; p1.c[0].virsta=5;
 cout << pp->c->virsta;
// 1
cout << pp->c[0].virsta;
// 2
cout << (pp->c+1)->virsta;
// 3
cout << ((*pp).c)->virsta;
// 4
cout << (*pp).c[1].virsta;
// 5
cout << (*pp).c->virsta;
// 6
}
```

Care din expresiile:

```
pp->c->virsta;              // 1
pp->c[0].virsta;            // 2
(pp->c+1)->virsta;          // 3
((*pp).c)->virsta;          // 4
(*pp).c[1].virsta;          // 5
(*pp).c->virsta;            // 6
```

sunt corecte?
a) toate;
b) 2+5;
c) 2+5+6;
d) 1+2+5+6;
e) 1+2+3+5+6;

6.

Descriptorul "*%+5d* " prezent într-un format
a) forteaza afisarea semnului, în cazul numerelor întregi pozitive;
b) serveste doar la editarea numerelor întregi pozitive;
c) serveste la editarea numerelor întregi, în modul;
d) serveste la cadrarea la dreapta a textului afisat.
e) inhiba afisarea semnului, când numarul este pozitiv;

7.

Care este rezultatul evaluarii expresiei *x = x&&y||z;* în ipoteza ca *x=2* si *y=0*; (*x,y,z* - sunt întregi).

a) depinde de valoarea initiala a lui z;
b) 1;
c) 0;
d) false, expresia fiind neadmisa în limbaj;
e) alta valoare decît cele propuse.

8.

Indicati posibilitatea evaluarii expresiei si eventual rezultatul acestei evaluari:
x = (x||!y)&&z; daca initial *x=2, y=1, z=0;*

a) 0;
b) expresia este eronata si nu este admisa în limbaj;
c) 1;
d) 2;
e) nici una din variantele propuse.

9.

Dupa declaratia: *char test[10]="Examen";*
sizeof(test):
a) returneaza 10;
b) returneaza 7;
c) returneaza 6;

d) nu se aplica la siruri, aici operînd functia *strlen()*;
e) returneaza 2, deoarece test, ca nume de masiv, este un pointer near.

10.

```
#include <stdio.h>
#include <string.h>
void main()
{ static char s1[100]=" daca ";
 static char s2[50]=" inveti", s3[50]=" raspunzi";
  strcat(s1,s2); strcat(s3,s1);
  puts(s3);
 }
```

a) afiseaza *raspunzi daca inveti*;
b) afiseaza *daca inveti raspunzi*;
c) este gresit, deoarece prin concatenare suprascrie zone de memorie nealocate la dimensiunea necesara;
d) foloseste variabile ce nu pot fi manipulate ca sir;
e) afiseaza *inveti daca raspunzi*;

11.

Fie programul:

```
#include <conio.h>
#include <iostream.h>
int &f(int &i){ i++; return i;}
void main( ){int c=7, a; a = ++f(c);
cout << endl << a <<“ “  << c; getch(); }
```

Valorile afisate pentru variabilele *a* si *c* sunt:
a) 9 si 9;
b) 7 si 8;
c) 8 si 9;
d) 8 si 8;
e) 8 si 10.

12.

Fiind data urmatoarea secventa:

```
#include <iostream.h>
class c
{ int a;
   public :
   void cit(){cout << "cit" << endl;};
void prel(){ cout << "prel" << endl;};
void rez(){cout << "rez" << endl;};
} ;
void main()
{ c ob;
   void (c::*v[ ]) ( )={c::cit, c::prel, c::rez };
   void ( c::* (*x)[3] )();
   x=&v; (ob.*x[0][2])();
}
```

În declaratia *void (c::*(*x)[3])()*;, x reprezinta:
a) pointer la vector de pointeri la metode ale clasei c;
b) pointer la o metoda a clasei c;
c) vector de pointeri la metode ale clasei c;
d) vector de pointeri la functii ce returneaza obiecte de tip c;
e) pointer la functie ce primeste masiv de obiecte de tip c si returneaza void.

13.

Fie data urmatoarea ierarhie

```
class B  {       /*......*/.          } ;
class D1:B       {  /*......*/       } ;
class D2:B       { /*......*/        } ;
class M1:D1, public D2  {/*......*/     };
class M2:virtual D1, virtual public D2
          {/*......*/     };
```

Precizati care afirmatii sunt corecte:
1) clasa M1 va mosteni un obiect de tip B;
2) clasa M1 va mosteni doua obiecte de tip B;
3) clasa M2 va mosteni un obiect de tip B;
4) clasa M2 va mosteni doua obiecte de tip B;

a) 2+3
b) 1+2
c) 2+4
d) 1+3
e) 1+4

14.

```
#include <iostream.h>
class vector
{ int *pe , nr_c ;
   public: operator int (){ return nr_c;}
             vector(int );  };
```

```
vector:: vector(int n)
{ pe=new int[n];nr_c=n;
  while(n--) pe[n]=n;
 }
void f( int i){ cout << i<< endl; }

void main()
{ vector x(10); f(x); }
```

Programul afiseaza:

a) 10 ;
b) 9;
c) numerele de la 1 la 10;
d) numerele de la 0 la 10;
e) numerele de la 0 la 9.

15.

```
#include <iostream.h>
void main()
{ int x=1, &r=x, z=3, y=2;
  cout << endl  << r  << endl;
  r = y; cout << r << endl;
}
```

Programul:

a) afiseaza 1, 2
b) afiseaza 1,1
c) afiseaza 2,2
d) foloseste incorect o referinta;
e) foloseste incorect un operator.

16.

Se considera urmatoarea secventa de co-enzi FoxPro:

```
SELECT 4
USE T2.DBF IN 2 ORDER 2
USE T3.DBF IN 3 INDEX C1, C2
```

Secventa de comenzi nu are ca efect:

a) Crearea unui index simplu pe coloanele C1 si C2 pentru tabela T3;
b) Deschiderea tabelelor T1 si T2;
c) Schimbarea zonei curente de lucru;
d) Activarea celui de-al doilea tag din fisierul index structural compus T2.CDX;
e) Selectarea zonei 4 ca zona curenta de lucru.

17.

Se considera tabela FIRME=(Cod, Denuire, Localitate, Sold_Plata)

Care este efectul executiei în FoxPro a urmatoarei cereri:

```
USE FIRME
GO 10
LIST OFF ALL FOR Sold_Plata > 0
WHILE Localitate='Brasov'
```

a) Afiseaza firmele care au Sold_Plata>0 începând cu înregistrarea 10 atâta timp cât Localitate='Brasov';
b) Afiseaza firmele care au Sold_Plata>0 si Localitate='Brasov', mai putin primele 10 firme;
c) Afiseaza toate firmele care au Sold_Plata>0 si Localitate='Brasov';
d) Afiseaza din primele 10 firme pe cele care au Sold_Plata>0 si Localitate ='Brasov';
e) Afiseaza toate firmele care au Sold_Plata>0.

18.

Comanda FoxPro:

```
SEEK 1
```

regaseste înregistrarile dintr-o tabela în mod:

a) Direct dupa cheie;
b) Direct dupa numarul înregistrarii;
c) Secvential;
d) Dinamic;
e) Nu este o comanda de regasire.

19.

Care din urmatoarele secvente de cautare a procedurilor este utilizata în FoxPro:

1. în acelasi fisier cu programul principal;
2. ca fisier separat pe disc;
3. în fisierul de proceduri specificat în co-anda de apel a procedurii;
4. în fisierul de proceduri activat de ultima comanda SET PROCEDURE;

a) 3,1,4,2;
b) 1,2,3,4;
c) 1,3,4,2;
d) 1,4,3,2;
e) 2,1,4,3;

20.
Se considera urmatoarea secventa de comenzi FoxPro:
USE CAMERE TAG Tip_Camera
SCAN
Tip=Tip_Camera
SCAN WHILE Tip_Camera=Tip
DISPLAY
ENDSCAN
ENDSCAN
Care din urmatoarele afirmatii este adevarata:
a) Secventa afiseaza toate înregistrarile din tabela CAMERE, grupate pe tipul camerei, mai putin prima camera din fiecare tip cu exceptia primului tip.
b) Secventa este gresita pentru ca buclele SCAN nu contin comanda de salt la urmatoarea înregistrare (SKIP 1).
c) Secventa afiseaza toate înregistrarile din tabela CAMERE, grupate pe tipul camerei.
d) Secventa este gresita pentru ca buclele SCAN nu contin conditia WHILE NOT EOF().
e) Secventa afiseaza toate înregistrarile din tabela CAMERE, grupate pe tipul camerei, mai putin ultima camera din fiecare tip.

21.
În cazul metodologiei SSADM, în etapa de proiectare a noului sistem, intrarile si iesirile pentru noul sistem se vor identifica din:
a) diagrama contextuala (nivelul 0) a modelului logic al sistemului proiectat;
b) diagrama contextuala (nivelul 0) a modelului logic al sistemului existent;
c) diagrama de flux a datelor, nivelul frunza, a modelului fizic al sistemului existent;
d) diagrama de flux a datelor, nivelul frunza, a modelului logic al sistemului proiectat;
e) catalogul cerintelor.

22.
Conform metodologiei SSADM, modelul logic al sistemului proiectat se obtine pe baza:
a) cerintelor functionale si a modelului logic al sistemului existent;
b) catalogului cerintelor;
c) modelului fizic al sistemului existent;
d) modelului logic al sistemului existent;
e) cerintelor nefunctionale si a modelului logic al sistemului existent;

23.
Pentru codificarea materiilor prime si materialelor, care din urmatoarele tipuri de coduri sunt de preferat:
a) coduri secventiale cu formare de grupe;
b) coduri matriceale;
c) coduri secventiale;
d) coduri binare;
e) coduri ierarhice.

24.
Fiind data urmatoarea structura a unei baze de date:
MATERIALE=(CodMat, DenMat, Um)
MATERIALE_CONTRACTATE=(NrContr, CodMat, PU)
ESALONARI_LIVRARI=(NrContr, CodMat, Luna, CantContractata, CantLivrata)
CONTRACTE=(NrContr, CodFurniz, DataContr)
FURNIZORI=(CodF, CodLoc, DenF, Telefon, ContBanca)
LOCALITATI=(CodLoc, DenLoc, PrefixTel)
Considerati ca structura este:
a) corect definita;
b) incorect definita datorita dependentelor functionale multivaloare existente;
c) incorect definita datorita existentei unor redundante;
d) incorect definita datorita existentei unor dependente tranzitive;
e) incorect definita datorita cheilor primare alese.

25.
Tehnica concordantei intrari-iesiri privind analiza si proiectarea sistemelor informatice nu ofera posibilitatea pentru:
a) definirea obiectivelor sistemului informatic;
b) proiectarea iesirilor sistemului informatic;
c) proiectarea intrarilor sistemului informatic;
d) definirea colectiilor de date;
e) corelarea iesirilor cu intrarile sistemului.

26.
Fiind data urmatoarea structura a unei baze de date:
MATERIALE=(CodMat, DenMat, UM, PretMediu, StocExistent, StocNormat, StocSiguranta, CodGestiune)
GESTIUNI=(CodGest, DenGestiune, NumeGestionar, GarantieMateriala)
CONTRACTE=(NrContr, Data, CodFurnizor)
MATCONTR=(NrContr, CodMat, PU)
ESALONARI=(NrContr, CodMat, Luna, CantContr, CantLivr)
FURNIZORI=(CodF, DenF, Adresa, ContBanca, CodFiscal, Telefon, Fax)
Care dintre urmatoarele situatii de informare-raportare pot fi obtinute pe baza acestei structuri:
1. Situatia imobilizarilor de mijloace circulante;
2. Situatia intârzierilor la livrare pe materii prime si materiale;
3. Situatia stocurilor existente de materii prime si materiale;
4. Situatia stocurilor de materii prime si materiale pe gestiuni;
5. Situatia cantitatilor contractate de materii prime pe furnizori si materiale;
6. Situatia materiilor prime cu miscare lenta si fara miscare;
7. Situatia necesarului de materii prime de aprovizionat;
8. Rulajul intrarilor si iesirilor de materiale;
9. Situatia materiilor prime pentru care s-a incheiat cel putin un contract de aprovizionare;
10. Situatia decontarilor cu furnizorii;
a) 1, 2, 9;
b) 1, 4, 6;
c) 2, 5, 7;
d) 3, 4, 8;
e) 5, 9, 10.

27.
Prin proiectarea structurii unei baze de date relationale se urmareste sa se asigure:
1. flexibilitatea tabelelor;
2. stabilitatea tabelelor;
3. simplitatea tabelelor;
4. înlaturarea anomaliilor de ordin logic;
5. facilitati de interfata cu utilizatorii;
6. utilizarea unor instrumente de tip CASE;
7. rationalizarea si simplificarea fluxurilor informationale;
8. implementarea unor modele matematice în cadrul sistemului informatic;
9. însusirea facila a exploatarii bazei de date de catre utilizatorii finali;
10. încarcarea masiva a bazei de date.
Care dintre urmatoarele variante este corecta:
a) 1, 2, 3, 4;
b) 1, 5, 7, 8;
c) 2, 3, 4, 6;
d) 4, 7, 8, 9;
e) 1, 3, 6, 10.

28.
Ce criterii se au în vedere în etapizarea activitatilor de realizare a sistemelor informatice:
a) diferitele categorii de personal antrenate în activitatea de realizare a sistemelor informatice precum si omogenitatea activitatilor de realizat;
b) diferitele categorii de personal antrenate în activitatea de realizare a sistemelor informatice;
c) omogenitatea activitatilor de realizat;

d) omogenitatea activitatilor si fluxul thenologic de prelucrare a datelor;
e) nici un criteriu.

29.
Care din urmatoarele operatii se pot realiza cu ajutorul limbajului SQL*PLUS:
1. definirea obiectelor din baza de date;
2. definirea restrictiilor de integritate;
3. definirea procedurilor si functiilor;
4. definirea drepturilor de acces;
5. structuri repetitive de prelucrare;
6. prelucrari la nivel de set de înregistrari;
7. structuri alternative de prelucrare;
8. tratarea erorilor.

a) 2, 4, 6;
b) 3, 4, 5;
c) 1, 3, 7;
d) 5, 6, 7;
e) 1, 4, 8.

30.
Notiunile utilizate în definirea structurii relationale a datelor sunt:
a) atribut, schema relatiei, chei, domeniu, relatie, tuplu;
b) relatie, fisier, tuplu, înregistrare, tip set;
c) tabela, atribut, tuplu, tip înregistare, chei
d) tabela, fisier, cîmp, valori, tuplu;
e) diferenta, disjunctia, conjunctia, diviziuea;

PROBLEMA

Pe un suport extern de memorie, se dau urmatoarele informatii despre activitatea de personal:
- marca
- nume si prenume
- vîrsta
- profesia
- vechimea în munca
- salariu tarifar

Sa se determine:
- salariul mediu pe grupe de vîrsta: pîna la 35 ani, între 35-50 ani si peste 50 ani;
- suma totala platita ca spor de vechime, la nivelul întregii unitati, stiind ca sporul procentual de vechime pe intervale de vechime este: 1-5 ani 3%, 5-10 ani 5%, 10-20 ani 9%, 20-30 ani 10 %, peste 30 ani 15 %)
- salariul mediu pe fiecare profesie distincta.

Observatie:
Fiecare întrebare se noteaza cu 2 puncte, iar problema cu 30 puncte.