BABEŞ-BOLYAI TUDOMÁNYEGYETEM

MATEMATIKA ÉS INFORMATIKA KAR

Záróvizsga – 2015 szeptember
Magyar informatika szak

1. tétel

Írjunk egy programot a Python, C++, Java, vagy C# nyelvek egyikében, melyben:

(a) egy *Gyógyszer* **osztályt definiál**unk egy valós típusú *ár* privát attribútummal, egy publikus konstruktorral, mely az *ár*at inicializálja, és egy *eladásiÁr*() publikus metódussal, mely a gyógyszer *árát* téríti vissza.

(b) egy *TámogatottGyógyszer* **osztályt definiál**unk, mely a *Gyógyszer* osztály leszármazottja, egy valós típusú *támogatásiArány* privát mezővel rendelkezik (mely a gyógyszer támogatásának az arányát fejezi ki). Ezen kívül az osztály rendelkezik egy publikus konstruktorral, mely az *ár* és *támogatásiArány* mezőket inicializálja, valamint egy publikus felülírt *eladásiÁr*() metódussal, mely a támogatott árat téríti vissza.

(c) egy **függvényt definiálunk**, mely egy gyógyszereket tartalmazó listát épít fel és térít vissza. A lista első eleme *Gyógyszer* típusú, melynek ára 100, a második eleme *TámogatottGyógyszer* típusú, melynek *ára* 70 és *támogatási aránya* 0.05, illetve egy szintén *TámogatottGyógyszer* típusú harmadik objektumot, melynek *ára* 90 és *támogatási aránya* 0.8.

(d) egy **függvényt definiálunk**, melynek bemenő paraméterei egy gyógyszereket tartalmazó lista – a (c) ponthoz hasonlóan – illetve egy *v* valós szám. A függvény a lista azon elemeit rendezi az eladási ár szerint csökkenő sorrendbe, melyek eladási ára nagyobb, mint *v*. A *v* értéknél kisebb gyógyszerek pozíciója nem változik az eredeti listában.

(e) egy **függvényt definiál**, melynek bemenő paramétere egy gyógyszer-lista – a (c) ponthoz hasonlóan – és a függvény a listából törli azon elemeket, melyek eladási ára kisebb, mint 40.

(f) **definiálja a főprogramot**, mely a (c) pontbeli listát építi fel, meghívja a (d), majd az (e) pontbeli függvényeket, majd a standard kimenetre kiírja a listában maradt gyógyszerek eladási árát.

(g) specifikáljuk az általunk alkalmazott lista műveleteket.

Megjegyzések:

- Ne használjunk rendezett konténereket;
- Ne használjunk a fentieken kívüli metódusokat a program megírásánál;
- Ne használjunk előre definiált rendező függvényeket.

*A **Lista** típusnál használhatunk létező függvénykönyvtárakat (Python, C++, Java, C#). Amennyiben nem standard könyvtárakat használunk, írjuk le az összes felhasznált műveletet.*

2. tétel

a. Tervezzünk **relációs adatbázis sémát,** melynek táblái **3NF**-ban vannak az Untold fesztivál következő infomációira:

- színpadok: színpad id, név, cím;
- művészek: művész id, név, származási ország, zene stílus (stílus id, név, leírás), alakulási év, színpad, ahol fellép, koncert napja, kezdeti időpont. Egy művész egyszer lép fel a fesztiválon.
- jegyek: jegy kód, jegy típusa (típus id, ára, megnevezése: *early bird*, *full price* vagy *pay with blood*), vásárló életkora, művészek listája, akiknek a koncertjén részt vettek a jeggyel.

Indokoljuk, hogy a táblák 3NF-ben vannak, funkcionális függőségeket használva.

b. SELECT-SQL parancsot vagy relációs algebrát használva, az **a.** pont adatbázisára vonatkozóan adjuk meg:

b1. A fesztivál programját (művész név, koncertje kezdeti időpontja) 2015 augusztus 1.-én a kolozsvári Arénában.

b2. Azon színpadokat (színpad név), ahol *electro hip hop* zenét játszottak **és** ahol voltak olyan koncertek, melyeken részt vett legalább egy 70 évnél idősebb néző.

b3. Azokat a művészeket (művész név), akik előadásán a legtöbb *pay with blood* típusú jeggyel rendelkező 18-24 év közötti résztvevő volt jelen.

3. tétel

3.1 Feltételezzük, hogy az alábbi program minden utasítását helyesen hajtja végre a rendszer, és a pipe-ok helyesen vannak bezárva. Válaszoljunk az alábbi kérdésekre:

```
 1  int main() {
 2    int f, r, x, p[2];
 3
 4    x = getpid();
 5    pipe(p);
 6    f = fork();
 7
 8    if (x == getpid())             // 1. if
 9      close(p[1]);
10    if (f == 0 && x == getpid()) { // 2. if
11      close(p[0]);
12      write(p[1], &x, sizeof(x));
13    }
14    if (f > 0)                     // 3. if
15      read(p[0], &r, sizeof(r));
16    if (getppid() == x && f > 0) { // 4. if
17      close(p[1]);
18      read(p[0], &r, sizeof(r));
19      printf("%d\n", r);
20      exit(0);
21    }
22    if (x == getppid()) {          // 5. if
23      close(p[0]);
24      f++;
25      write(p[1], &f, sizeof(f));
26      exit(0);
27    }
28    if (getppid() == 0)            // 6. if
29      printf("%d\n", f);
30
31    printf("%d\n", r);
32  }
```

a) Mit jelent az **x** és **f** változók értéke?

b) Magyarázzunk meg részletesen minden **if** utasítást és a hozzájuk tartozó blokkokat.

c) Mi jelenik meg a standard kimeneten a program végrehajtása után? Miért?

3.2 Adott az alábbi UNIX shell script:

```
 1  rm tmp
 2  echo -n > tmp
 3  for f in $*
 4  do
 5    if test ! -f $f
 6    then
 7      echo $f nem letezik mint file
 8      continue
 9    fi
10    rm $f
11    if [ ! -f $f ]
12    then
13      echo $f sikeresen letorolve
14    fi
15    ls $f >> tmp
16  done
17  x=`cat tmp | grep -c ^.*$`
18  echo eredmeny: $x
```

a) Magyarázzuk meg mi a különbség az 5. sorban szereplő **if** utasítás, illetve a 11. sorban szereplő **if** utasítás között.

b) Magyarázzuk el részletesen a 17. sort.

c) Mit jelent (magyarázzuk is meg miért) a program végén kiírt **x** változó értéke?

d) Írjuk át a 10-14 programsorokat úgy, hogy a végrehajtás eredménye ugyanaz maradjon, de egy UNIX paranccsal kevesebbet használjunk.

Megjegyzés: Az összes tétel kötelező.
Minden tételt 1-től 10-ig osztályoz két javító.
Munkaidő: 3 óra.

BAREM

INFORMATICĂ

Subiect 1 (Algoritmică şi Programare):
Oficiu – **1**p
Definirea clasei *Medicament*– **0.75**p din care
- atribut – 0.25
- constructor – 0.25
- metoda *preţVânzare*() - 0.25

Definirea clasei *MedicamentCompensat*– **1.75**p din care
- relația de moștenire – 0.25
- constructor – 0.5
- atribut – 0.25
- metoda *preţVânzare*() – 0.75

Funcția de la punctul c) – **1**p din care
- signatura corectă și declarare listă- 0.25p
- creare obiecte – 0.25p
- adăugare obiecte în listă - 0.25p
- returnare rezultat - 0.25p

Funcția de la punctul d) – **1.5**p din care
- signatura corectă - 0.25p
- sortare listă conform cerințelor – 1p
- returnare rezultat - 0.25p

Funcția de la punctul e) – **1.5**p din care
- signatura corectă - 0.25p
- parcurgere listă și ștergere elemente cerute – 1p
- returnare rezultat - 0.25p

Program – **1**p din care
- apel funcții – 0.25p
- afișarea prețuri din listă – 0.75p

Specificațiile operațiilor folosite din tipul de dată *Listă*– **1.5**p

Subiect 2 (Baze de date):
1 punct oficiu

a) 2p justificare
   2p tabele corecte in 3NF

b) b1 - 1p
   b2 - 2p
   0.5p pentru scenele cu muzica electro hip hop
   0.5p pentru scenele care au gazduit concerte cu cel putin un participant peste 70 ani
   1p instructiunea finala
   b3- 2p
   1p grupare dupa artisti, calculare numar participanti cu conditie
   1p instructiunea finala

Subiect 3 (Sisteme de operare):

Oficiu: 1p

3.1

a) 0.5p valoarea variabilei **x**, 0.5p valoarea variabilei **f**

b) 0.5p x 6 **if**-uri

c) Se afişează valoarea 1 + explicație de ce: 1p

3.2

a) nu sunt diferite 1p

b) explicații detaliate 1p

c) numărul de fişiere care nu au putut fi şterse 1p

d) `if rm $f`, se elimină astfel folosirea comenzii `test` sau `[` 1p