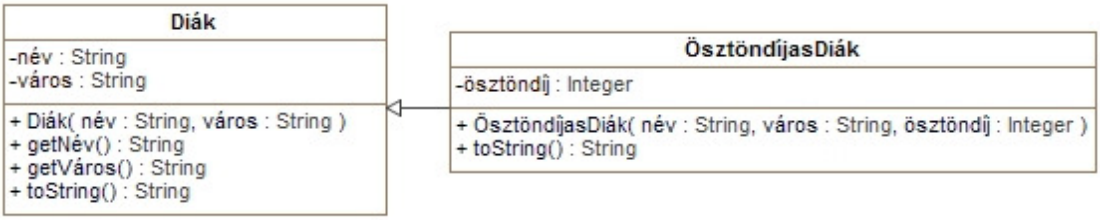
BABEŞ-BOLYAI TUDOMÁNYEGYETEM

MATEMATIKA ÉS NFORMATIKA KAR

Licenszvizsga, 2016 június
Informatika – magyar nyelv

1. tétel: Írjunk programot a C++, C#, Java, Python programnyelvek valamelyikében, mely:

a) Definiálja a *Diák* és az *ÖsztöndíjasDiák* osztályokat az alábbi UML-diagramm alapján



- A *név*, valamint a *város* mezők értéke nem lehet null vagy üres; az *ösztöndíj* értéke pozitív kell, hogy legyen;
- Az alaposztály *toString*() metódusa a diák *nev*ét és *város*át téríti vissza összefűzve; ugyanez a metódus a származtatott osztálynál felül lett írva úgy, hogy hozzáfűzi az *ösztöndíj* értékét, a *nev*et és *város*t tartalmazó String-hez.

b) Definiál egy függvényt, mely egy *Diák* típusú objektumot beszúr egy – *név* szerint növekvő sorrendbe rendezett – *Diák*-listába úgy, hogy a lista rendezett marad;

c) Definiál egy függvényt, mely a konzolra kiírja a diákok listáját;

d) Definiál egy függvényt, melynek bemenő paramétere egy Diák-lista; a függvény minden városra kiírja – rendezetten – a városbeli diákok listáját úgy, hogy mindegyik várost csak egyszer írunk ki.

e) A main() függvény a (b) pontban adott függvény segítségével felépít egy négyelemű listát, ahol a diákok városai „Nagyvárad” és „Arad”, háromnak van ösztöndíja. A lista felépítése után kiírja a diákok listáját a (c) pontbeli függvénnyel, majd meghívja a (d) függvényt.

f) Adjuk meg a programban használt adattípusok műveleteinek a specifikációját.

Megjegyzések: Lehet használni a programnyelvek adatstruktúráit és könyvtárait (C++, C#, Java, Python). Ne használjunk rendezett adattípusokat vagy beépített rendező műveleteket

2. tétel:

a) Tervezzünk relációs adatbázis sémát, melynek táblái 3NF-ban vannak egy online turisztikai foglalási cég infomációira:

b1) turisták: név, email, helységkód, helységnév, országkód, országnév;

b2) szálláshelyek: név, helységkód, helységnév, országkód, országnév, szállástípus kódja, szállástípus neve (lehetséges értékek: panzió, hostel, szálloda, *stb.*), csillagok száma, értékelés, ár/éj, valamint a foglalások listája, mely tartalmazza a turistát, a foglalás kezdeti dátumát, illetve a napok számát.

A funkcionális függőségeket megadva, indokoljuk meg, hogy a táblák 3NF-ben vannak.

b) SQL parancsot vagy relációs algebrát használva, az a) pont adatbázisára vonatkozóan adjuk meg:

b1) Azokat a turistákat (név, email), akik foglaltak legalább egy 9-nél nagyobb értékelésű panziót, de nem foglaltak egyetlen 9-nél kisebb értékelésű szállodát sem.
b2) Kolozsvári turistáknak 5 csillagos párizsi szállodákban való összes foglalásának a számát.
b3) Azokat a szálláshelyeket (név, helységnév, szállás típus név, csillagok száma), ahol az utóbbi 5 évben a legtöbb foglalás volt.

3. tétel

3.1 Adott az alábbi kódrészletet tartalmazó grep.c állomány; a felhasználó alapkatalógusában lefordított végrehajtható állomány neve pedig grep. Feltételezve, hogy minden utasítás hiba nélkül fut le, válaszoljunk az alábbi kérdésekre:

```
1  int main(int c, char** v) {
2    int p[2], n;
3    char s[10] = "valami";
4    pipe(p);
5    n = fork();
6    if(n == 0) {
7      close(p[0]);
8      printf("elotte\n");
9      if(c > 2)
10       execlp("grep","grep",v[1],v[2],NULL);
11     strcpy(s, "utana");
12     write(p[1], s, 6);
13     close(p[1]);
14     exit(0);
15   }
16   close(p[1]);
17   read(p[0], s, 6);
18   close(p[0]);
19   printf("%s\n", s);
20   return 0;
21 }
```

a) Soroljuk fel és magyarázzuk meg az n változó lehetséges értékeit.

b) Mi jelenik meg a képernyőn az alábbi futtatások esetében, tudva, hogy a felhasználó alapkatalógusa nem szerepel a PATH környezeti változóban.

b.1) **`grep grep grep.c`**
b.2) **`./grep grep grep.c`**
b.3) **`./grep grep`**

3.2 Adott az alábbi UNIX shell scriptet tartalmazó abc.sh állomány. Válaszoljunk az alábbiakra:

```
1  n=0
2  for i in `cat $1`; do
3    c=`echo $i|cut -c1`
4    if echo $i|grep -q "^[0-9][0-9]*$"; then
5      echo $i >> $1.nr
6    elif echo $c|grep -q "[A-Za-z]"; then
7      echo $i >> $c
8    else
9      n=`expr $n + 1`
10   fi
11 done
12 echo $n
```

a) Magyarázzuk el a 4. sorban található reguláris kifejezés jelentését.
b) Mi történik, ha a scriptet paraméter nélkül hívjuk meg?
c) Mi kerül a képernyőre a **`./abc.sh f3`** futtatása nyomán, és milyen új állományokat hoz létre (név és tartalom), amennyiben az **`f3`** tartalma **`"abc 74 2-8 aa 3a =c b2"`**, a futtatás pedig egy olyan katalógusban történik, ami csak az **`abc.sh`** és **`f3`** állományokat tartalmazza?
d) Adjunk példát egy **`f3`** állományra úgy, hogy az előző pontban megadott futtatás eredményeként 4 új állomány jöjjön létre, melyek közül egyik neve sem kezdődik az **`f3`** előtaggal.

Megjegyzés: Minden tétel kötelező. Minden tételt két javító osztályoz 1 és 10 közötti jeggyel.
Munkaidő: 3 óra.

BAREM

INFORMATICĂ

Subiect 1 (Algoritmică şi Programare):
Oficiu – **1**p
Definirea clasei Student– **0.75**p din care
- atribute – 0.25
- constructor – 0.25
- metode - 0.25

Definirea clasei StudentBursier– **1.25**p din care
- relația de moștenire – 0.25
- constructor – 0.5
- metoda toString() – 0.5

Funcția de la punctul b) – **2**p din care
- signatura corectă - 0.1p
- algoritmul de inserare în interiorul listei - 1.8p
  - parcurgere listă și determinarea poziției de inserare – 1.6p
  - adăugare element pe poziția determinată anterior – 0.2
- returnare rezultat - 0.1p

Funcția de la punctul c) - **0.5**p din care
- signatura corectă - 0.1p
- parcurgere listă și afișare – 0.4p

Funcția de la punctul d) – **2.5**p din care
- determinare lista bursieri din orașe – 2p
- tipărire perechi – 0.5p

Funcția de la punctul e) – **0.5p**
Specificaţiile operaţiilor tipurilor de dată folosite– **1.5**p

Subiect 2 (Baze de date)

1 punct oficiu

Problema a:

1 punct pentru dependenţe funcţionale

2 punct pentru tabelele în 3NF;

1 punct pentru justificare.

Problema b:

1 puncte pentru b1

1.5 puncte pentru b2

2.5 puncte pentru b3

Subiect 3 (Sisteme de operare):
Oficiu: 1p
3.1
- a) 1p - Valorile posibile
  1p - Explicații
- b.1) 1p - Afișează linia 10
- b.2) 1p - Afișează "înainte", linia 10 și "ceva"
- b.3) 1p - Afișează "înainte" și "după"

3.2
- a) 1p - Secvență una sau mai multe cifre
- b) 0.5p - Comanda "cat" așteaptă input de la intrarea standard
- c) 0.5p - Afișează 3 pe ecran
  0.5p - Fișierul f3.nr conținând 74
  0.5p - Fișierele dicționar a și b, conținând "abc aa" respectiv "b2"
- d) 1p - Orice secvență de cuvinte dintre care 4 încep cu litere diferite