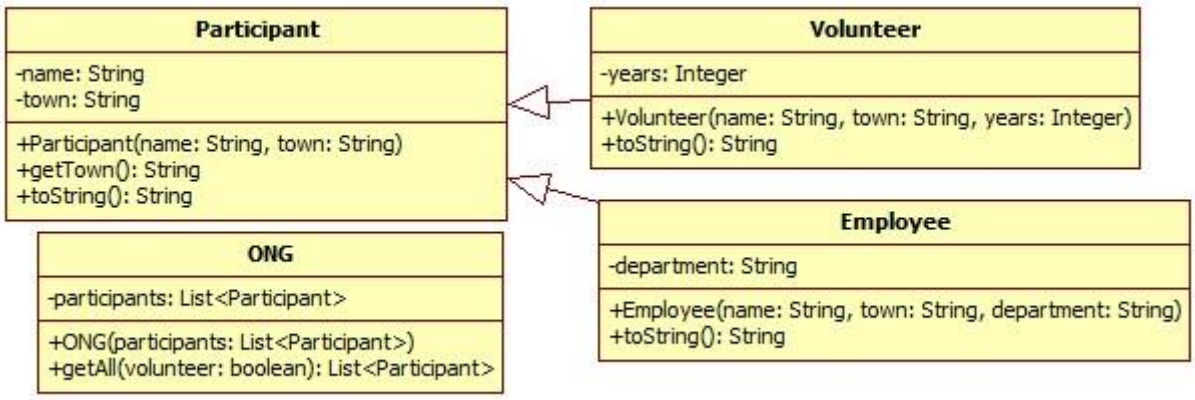
**BABEŞ-BOLYAI TUDOMÁNYEGYETEM**
**MATEMATIKA ÉS INFORMATIKA SZAK**

# Licenszvizsga, 2018 szeptember 4
# Informatika - magyar nyelv
# 2. VÁLTOZAT

## 1. tétel Algoritmusok és programozás

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

a) **definiálja** a **Participant** (résztvevő), **Volunteer** (önkéntes), **Employee** (alkalmazott), valamint az **ONG** (civilszervezet) osztályokat a következő UML-diagram alapján:




- A *name* (résztvevő neve), *town* (résztvevő városa) és a *department* (részleg) nem null-értékűek és nem üresek; a *years* (évek) szigorúan pozitív. A megszorításokat a konstruktorok validálják.
- A **Participant** osztály **toString()** metódusa a *name* és *town* mezők összefűzését téríti vissza. A **Volunteer** osztály **toString()** metódusa által visszatérített string a *"Volunteer <years> years"* és az alaposztály **toString()** visszatérési értékének az összefűzése; az **Employee** osztály **toString()** metódusa pedig az "*Employee <department>*" és az alaposztály **toString()** értékének az összefűzése.
- Az **ONG** osztály **participants** mezője **Participant** típusú objektum-lista; az **ONG** osztály **getAll(***volunteer*:boolean**)** metódusa az önkéntesek (**Volunteer**) vagy az alkalmazottak (**Employee**) listáját téríti vissza a *volunteer* paraméter értékének függvényében.

b) **Definiál egy függvényt**, mely az **ONG** típusú bemeneti objektumra, rendezi és visszatéríti az adott civilszervezetben szereplő önkéntesek (**Volunteer)** listáját; a rendezés a származási *város (town)* szerint történik ábécé szerinti növekvő sorrendben. A rendezésnél használjuk az összefésüléses rendezést - Merge Sort.

c) **Definiál egy függvényt**, melynek bemeneti paramétere egy **ONG** objektumokat tartalmazó lista és visszatéríti a legtöbb alkalmazottal (**Employee**) rendelkező várost. Több ilyen város esetén a függvény az egyiket téríti vissza.

d) **Definiál egy függvényt**, mely a bemeneti, **ONG** elemeket tartalmazó, listára visszatéríti az alkalmazottak (**Employee**) számát.

e) **Definiálja** a program fő-függvényét, mely létrehoz egy kételemű, **ONG** objektumokat tartalmazó listát (a mezők értékei tetszőlegesek): az első két **Volunteer** és egy **Employee** típusú objektumot, a második pedig egy **Volunteer** és egy **Employee** típusú objektumot tartalmaz. Hívjuk meg a (b)-ben definiált függvényt az ONG lista második elemére, majd a (c) és (d) pontok függvényeit az ONG-k listára.

f) Adjuk meg a **Lista** adattípusra a használt műveletek specifikációját.

**Megjegyzés:**

- **Adjuk meg a használt programozási nyelvet.**
- **Az UML-diagram osztályainál ne használjunk a specifikáción kívüli függvényeket.**
- **Ne használjunk rendezett konténereket és létező rendezési műveleteket.**

*Az **adatstruktúrákhoz** használhatunk függvénykönyvtárakat (C++, Java, C#, Python).*

## 2. tétel Adatbázisok

Legyen egy adatbázis, mely az utóbbi 50 év személyszállító repülőjáratait tárolja. Az adatbázis szerkezete a következő:

- ***Légitársaságok*** tábla, attribútumai: **LTKód, Megnevezés, Régiség, Ország, Kontinens**;
- ***Repülőterek*** tábla, attribútumai: **RTKód**, **Név**, **Város, Ország, Kontinens**;

- ***Járatok*** tábla, attribútumai: **LégiTársaságKód**, **ReptérHonnanKód**, **ReptérHovaKód, Dátum**, **Időpont**, **RepülőgépTípus, HelyekSzáma**.

1. Határozzuk meg az elsődleges és külső kulcsokat!
2. Adjunk meg legalább 3 funkcionális függőséget, melyek nem kód-attribútumokra vonatkoznak.
3. A következő szerkezeti módosítások közül, melyek szükségesek ahhoz, hogy az adatbázis harmadik normál formában (**3NF**) legyen. Indokoljuk a választást!
   a) Az országok tárolására külön tábla létrehozása.
   b) **AlapításDátuma** attribútum használata a **Régiség** attribútum helyett.
   c) A repülőgép típusok tárolására külön tábla létrehozása.
   d) **ReptérHonnanKód** > **ReptérHovaKód** megszorítás létrehozása.
4. Írjunk SQL lekérdezést az eredeti szerkezetre vonatkozóan, mely ekvivalens a következő lekérdezéssel:

$$\Pi_{\text{Megnevezés}} ( \sigma_{\text{Kontinens = 'Ázsia'}} ( \textbf{Légitársaságok} \bowtie_{\text{LTKód = LégiTársaságKód}} \sigma_{\text{HelyekSzáma >200}} (\textbf{\textit{Járatok}}) ))$$

5. Írjunk egy SQL lekérdezést, mely minden repülőtér esetén megadja a be és kimenő járatok számát. (***Név, JáratokSzáma***).

## 3. tétel Operációs rendszerek

**3.1** Feltételezve, hogy az alábbi kódrészlet minden utasítása hiba nélkül hajtódik végre, válaszoljunk a következő kérdésekre:

```
1  int main(){
2    int pfd[2], i;
3    char buffer, c;
4    pipe(pfd);
5    for(i=0;i<3;i++){
6      if(fork()==0){
7         while(read(pfd[0],&buffer, 1)>0){
8           printf("%c\n", buffer);
9         }
10        close(pfd[0]);
11        close(pfd[1]);
12        exit(0);
13     }
14   }
15   close(pfd[0]);
16   for(i=0;i<10;i++){
17     c = 'a' + i;
18     write(pfd[1],&c,1);
19   }
20   close(pfd[1]);
21   while(wait(NULL)>0){}
22   exit(0);
23 }
```

a) Rajzoljuk le a létrejött folyamatok hierarchiáját, a szülő folyamatot is beleértve.

b) Mi lesz a kimenet a program végrehajtását követően?

c) Magyarázzuk meg, hogy miért nem fejeződnek be a gyerekfolyamatok.

d) Melyik sorok közé kéne a 11. kódsort költöztetni ahhoz, hogy a gyerekfolyamatok befejeződjenek?

e) Magyarázzuk meg a 21. kódsort.

**3.2** Az alábbi UNIX shell script az a.sh nevű állományba van elmentve. Válaszoljunk az alábbi kérdésekre, a következő parancs végrehajtását tekintve: `cat test | ./a.sh`

```
#!/bin/bash
read a
x=0
while [ "$a" != "" ]; do
  if echo "$a" | grep -q "^[0-9].*$"
  then
    x=`expr $a + $x`
  fi
  read a
done
echo $x
```

a) Mi lesz a kimenet, amennyiben a `test` állomány kizárólag a 0-tól 5-ig terjedő számokat tartalmazza, külön sorokban?

b) Mi lesz a kimenet, amennyiben a `test` állomány kizárólag egyetlen sort tartalmaz, melyben a 0-tól 5-ig terjedő számok szóközzel vannak elválasztva?

c) Mi lesz a kimenet, amennyiben az (a) pontnál megadott `test` állományhoz hozzáadjuk az `abc` karaktersort tartalmazó sort?

d) Mi lesz a kimenet, amennyiben az (a) pontnál megadott `test` állományhoz hozzáadjuk az `6bc` karaktersort tartalmazó sort?

**Megjegyzések.**

- Mindegyik tétel kötelező; a tételeknél teljes megoldásokat kérünk.
- A dolgozat minimális átmenő jegye az ötös (5,00).
- Munkaidő: három (3) óra.