ÉRETTSÉGI VIZSGA • 2010. május 14.

INFORMATIKAI ALAPISMERETEK

EMELT SZINTŰ ÍRÁSBELI ÉRETTSÉGI VIZSGA

JAVÍTÁSI-ÉRTÉKELÉSI ÚTMUTATÓ

OKTATÁSI ÉS KULTURÁLIS MINISZTÉRIUM

Fontos tudnivalók

Általános megjegyzések:

- Ha nem a kérdésben meghatározottak szerint válaszol, akkor a válasz nem fogadható el! (Pl.: **H** betű helyett nem válaszolhat **N** betűvel.)
- A feleletválasztásos tesztfeladatnál javítani tilos, a javított válaszok nem értékelhetők!
- Ha egy kérdésre a jó válasz(ok) mellett a vizsgázó válaszában hibás választ is megjelöl, akkor a kérdésre adható pontszámból le kell vonni a rossz válaszok számát. Negatív pontszám nem adható, ezért több hibás válasz esetén a minimális pontszám nullánál kevesebb nem lehet.

  Pl.: Ha egy jó válasz mellett a vizsgázó egy hibás választ is bejelöl, akkor 0 pontot kell adni. Ez nem vonatkozik azokra a kérdésekre, ahol a **(minden helyes részválasz 1 pont)** szöveg szerepel.
- A kifejtős kérdések (nem feleletválasztós) válaszainál nem a szó szerinti, hanem a helyes tartalmi, illetve a lényegi válaszok megadását kell értékelni. Ha a vizsgázó válaszának a tartalmi vonatkozásai megfelelnek a megoldási útmutatóban megadott válasznak, akkor a válaszra adható pontot meg kell adni. Ha csak kis részben, vagy pedig nem felel meg a kapott válasz, akkor pont nem jár a válaszért.
- A pontszámok az **I.** részben a megadott részletezésnél tovább nem bonthatók (0,5 pont nem adható).
- Egyes esetekben előfordulhat, hogy egy általánostól eltérő rendszer használata miatt valamely kérdésre a vizsgázó nem a várt válasz adja, de *a válasza és az indoklása elfogadható*. Ilyen esetben a kérdésre adható pontszámot meg kell adni.

  Pl.: Táblázatkezelőkben magyar beállításnál a tizedesek elválasztásának a jele a **vessző**, és ez a várt válasz. Ha a vizsgázók munkájuk során angol beállítást használnak, vagy a vizsgázó odaírja ezt megjegyzésként, akkor az előző helyett az angol beállítású környezetben használt **pont** lesz a helyes válasz.

A javítási-értékelési útmutatóban feltüntetett válaszokra kizárólag a megadott pontszámok adhatók.
A megadott pontszámok további bontása csak ott lehetséges, ahol erre külön utalás van. Az így kialakult pontszámok csak egész pontok lehetnek.

Egyszerű, rövid, illetve kifejtendő szöveges választ igénylő írásbeli feladatok.

Hardver

1) b. .....9600/10 = 960 bájt……2 pont

2) a.....1*768*4 = 3072 kbájt……...2 pont

3) c.....A benne lévő festékpor a nyomtatóhengernek a lézerfény által ért helyein megtapad, majd onnan a papírra hengerelődik……1 pont

4)
- A DVD-ROM lemezek optikai elvű adattárolók, gyárban írottak, újraírásuk nem lehetséges. Jellemzően szoftvereket, játékokat, multimédiás anyagokat tartalmaznak.
- A DVD-RAM lemezek magneto-optikai (magnet-optikai) elven tárolják az adatokat. A közönséges írható DVD-hez képest sokkal többször írhatóak……..2 + 2 = 4 pont

5) I, H, I, I……...4 x 1 = 4 pont

6) Veszteséges tömörítés……1 pont

7)d Mágneses adattároló……...1 pont

8) Neumann elvek:
- Kettes számrendszer használata
- Teljesen elektronikus működés
- Soros utasításvégrehajtás
- Központi vezérlőegység
- Belső memória a program és az adatok tárolására
- Univerzális működés

6 x 1 = 6 pont

Szoftver

9) Operációs rendszer……1 pont

10) H, H, I, H, I ……..5 x 1 = 5 pont

11) c.....dir>ment.txt……..2 pont

12) b.....Kártékony programok, a programfájlokat, a boot szektort, sőt a szövegszerkesztővel írt dokumentumokat is tönkretehetik, csak az operációs rendszer van védve tőlük……2 pont

Szövegszerkesztés, táblázatkezelés

13) H, I, I, H, H……5 x 1 = 5 pont

14) b.....Egy új szakaszt szúrunk be, majd megváltoztatjuk az oldal tájolását állóról fekvőre……..1 pont

Informatikai alapok

15) Vagy kapcsolat…….1 pont

16)

- az informatikában a kettes számrendszert alkalmazzák a könnyű fizikai megvalósíthatósága miatt
- a bináris számok könnyen átalakíthatók 16-os számrendszerbe, és viszont
- 16-os számrendszerben sokkal kevesebb számjeggyel írhatjuk le ugyanazokat az értékeket, tehát rövidített leírásra ad lehetőséget

3 x 1 = 3 pont

17) H, I, I……...2 + 1 + 2 = 5 pont

Hálózati alapismeretek, HTML

18)b 10000001 00101001 10000011 00110100……...2 pont

19) d <pre>KARCSI ALMÁT VESZ.</pre>……..2 pont

Összesen: 50 pont

Programozási feladatok számítógépes megoldása

1. feladat **10 pont**

A kitűzött feladat:

$\binom{n}{k}$ a kombinatorikában használatos függvény, értéke megadja, hogy n különböző elem közül hányféleképpen tudunk kiválasztani k darabot úgy, hogy a kiválasztott elemek sorrendje nem számít! A függvény értelmezéséből következik, hogy teljesülnie kell az $n \geq 0$ és a $n \geq k \geq 0$ feltételeknek!

$\binom{n}{k}$ meghatározására több különböző képlet ismert. A feladat megoldása során ezek közül a következőt kell alkalmaznia: $\binom{n}{k} = \frac{n \cdot (n-1) \cdot \ldots \cdot (n-k+1)}{1 \cdot 2 \cdot \ldots \cdot (k-1) \cdot k}$

Példa: $\binom{9}{4} = \frac{9 \cdot 8 \cdot 7 \cdot 6}{1 \cdot 2 \cdot 3 \cdot 4} = \frac{3024}{24} = 126$

Írjon programot, amely a felhasználó által megadott n és k értékek esetén meghatározza $\binom{n}{k}$ értékét a fentiekben leírt módszerrel!

- Az adatbevitel során a program külön-külön ellenőrizze mindkét bemenő adatot!
- Szükség esetén a program kérje be az adatot újra, mindaddig, amíg az nem teljesíti a feladatkitűzésben leírt feltételeket! Hibaüzenetet nem kell megjelenítenie.
- Típusellenőrzést nem kell végezni!
- Törekedjen arra, hogy a lehető legnagyobb n és k értékekkel tudjon számolni a program!

A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! Beadandó: a feladatot megoldó program forráskódja!

Mintamegoldás: a feladat egy lehetséges, C# nyelvű megoldása, megtalálható a *Feladat1.cs* állományban

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Feladat1
{
        class binomialis
        {
            public int n, k;
            public void beker()
            {
                do
                {
                    Console.Write(" Adja meg n értékét: ");
                    n = int.Parse(Console.ReadLine());
                }
                while (n <= 0);
                do
                {
                    Console.Write(" Adja meg k értékét: ");
                    k = int.Parse(Console.ReadLine());
                }
                while (!(n >= k && k >= 0));
            }


            private int binom(int n, int k)
            {
                double szamlalo = 1;
                double nevezo = 1;
                int j = n;
                for (int i = 1; i <= k; i++)
                {
                    szamlalo *= j--;
                    nevezo *= i;
                }
                return (int)Math.Round(szamlalo / nevezo);
            }

            public void binomkiir()
            {
                Console.WriteLine(" "+n+" alatt a " + k + ": " + binom(n, k));
            }
        }

        class Program
        {
            static void Main(string[] args)
            {
                binomialis b = new binomialis();
                b.beker();
                b.binomkiir();
                Console.ReadLine();
            }
        }

}
```

Értékelés:

a) A programkód szintaktikailag hibátlan, lefordítható ..... 1 pont
   - Ez a pont csak abban az esetben adható meg, ha a programkód tartalmaz a b-e szakaszokba tartozó, összességében legalább 3 pontot érő részmegoldást!

b) A változók helyes definiálása ..... 1 pont
   - A pont abban az esetben adható meg, ha a feladatmegoldáshoz szükséges valamennyi fő és segédváltozó deklarálásra került, valamint ha a részletszorzatok (azaz a számláló és a nevező) kiszámítása valós típusú változóban történik!

c) Adatbekérés, ellenőrzés ..... 3 pont
   - Ha mindkét bemenő adat bekérése megtörtént, a bekérés a felhasználó számára egyértelmű volt: 1 pont
   - n bekérése ellenőrzött, hátultesztelős ciklust alkalmaz, a feltétel helyes: 1 pont
   - k bekérése ellenőrzött, hátultesztelős ciklust alkalmaz, a feltétel helyes: 1 pont

d) A binomiális együttható meghatározása ..... 4 pont
   - Helyes ciklusszervezés (pl. számlálós ciklus 1-től k-ig): 1 pont
   - A számlálóban szereplő szorzat pontos meghatározása: 1 pont
   - A nevezőben szereplő szorzat pontos meghatározása: 1 pont
   - A hányados előállítása, érték kerekítése egésszé: 1 pont

e) Eredmény kiírása ..... 1 pont

2. feladat **10 pont**

A kitűzött feladat:

Egy fizikai kutatóintézetben gyakran végeznek olyan méréseket, amelyek kiértékelése során fontos szempont, hogy egy-egy érték hányszor fordul elő a méréssorozatban.

Készítsen programot, amely lehetőséget ad egy méréssorozat ilyen jellegű kiértékelésére. A program teljesítse a következőket:

- A program adjon lehetőséget a mért értékek beolvasására a billentyűzetről! Ezek tetszőleges valós számok lehetnek, de számuk legfeljebb 15 legyen.
- A beolvasás érjen véget, ha a felhasználó a „*” végjelet adja meg, vagy ha a beolvasott értékek száma elérte a 15-öt!
- Az adatbekérés során semmilyen egyéb ellenőrzést nem kell végezni!
- Az adatok beolvasása után a program jelenítse meg az egymástól különböző mért értékeket növekvő sorrendben, és mindegyik mellé írja oda az érték előfordulási gyakoriságát!
- A többször előforduló értékeket értelemszerűen csak egyszer kell kiírni!

Példa:
Tegyük fel, hogy a felhasználó a következő mérési eredményeket adja meg:

```
2,3   5,8   2,3   4,7   5,8   2,3
```

Ebben az esetben a kiértékelés:

```
2,3: 3 db    4,7: 1 db    5,8: 2 db
```

A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! Beadandó: a feladatot megoldó program forráskódja!

Mintamegoldás: a feladat egy lehetséges, C# nyelvű megoldása, megtalálható a *Feladat2.cs* állományban

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Feladat2
{
    class Felmeres
    {
        private const int max = 15;
        private double[] meresek = new double[max];
        private int n=0; // A mérések száma

        public void feltolt()
        {
            bool kilep=false;
            Console.WriteLine("=> Adatok beolvasása:");
            Console.WriteLine();
            do
            {
                Console.Write("  "+(n + 1) + ". mért érték: ");
                string s = Console.ReadLine();
                kilep=(s=="*");
                if (!kilep)
                   meresek[n++]= double.Parse(s);
            }
            while (n<max && !kilep);
        }

        public void rendez()
        {
            for (int i = 0; i < n-1; i++)
            {
                for (int j = i + 1; j < n; j++)
                {
                    if (meresek[i]>meresek[j])
                    {
                        double s=meresek[i];
                        meresek[i]=meresek[j];
                        meresek[j]=s;
                    }
                }
            }
        }

        public void kiir()
        {
            Console.WriteLine();
            Console.WriteLine("=> A mért értékek gyakorisága: ");
            Console.WriteLine();
            int db = 1;
            for (int i = 0; i < n; i++)
            {
                if ((i < n - 1 && meresek[i] < meresek[i + 1]) || (i==n-1))
                {
                    Console.Write("  "+meresek[i]+": "+db+" db");
                    db = 1;
                }
                else
                {
                    db++;
                }
            }
            Console.WriteLine();
            Console.ReadKey();
        }
    }
```

```
    class Program
    {
        static void Main(string[] args)
        {
            Felmeres m = new Felmeres();
            m.feltolt();
            m.rendez();
            m.kiir();
        }
    }
}
```

Értékelés:

a) A programkód szintaktikailag hibátlan, lefordítható ..... 1 pont
   - Ez a pont csak abban az esetben adható meg, ha a programkód tartalmaz a b-e szakaszokba tartozó, összességében legalább 3 pontot érő részmegoldást!

b) A konstansok és változók helyes definiálása..... 2 pont
   - A mérési eredmények tárolására alkalmas valós elemű tömb deklarálása: 1 pont
   - Az egyéb szükséges konstansok, változók helyes deklarálása: 1 pont

c) A mérési eredmények beolvasása..... 2 pont
   - A beolvasott mérési eredmények beolvasásra és tárolásra kerülnek, a mérési eredmények darabszámát a program meghatározza: 1 pont
   - A beolvasás véget ér, ha a '*' végjelet adja a felhasználó, vagy ha a tömb elemszáma elérte a 15-öt: 1 pont

d) Az adatok rendezése valamely ismert rendezési algoritmus segítségével..... 2 pont
   - Minden elemi hiba 1-1 pont levonást jelent, de negatív pontszám nem adható!

e) Előfordulási gyakoriságok kiírása ..... 3 pont
   - Helyes ciklus szervezés, minden mért érték pontosan egyszer kiírásra kerül: 1 pont
   - A mért értékek mindegyike megszámlálásra kerül: 1 pont
   - A mért értékek gyakorisága kiírásra kerül: 1 pont

3. feladat **15 pont**

A kitűzött feladat:

Karácsony közeledtével versenyt hirdetnek a fenyőfatermelők számára, „Ki adja az ország karácsonyfáját?” címmel. A verseny győztese szállíthatja a Parlament előtt felállítandó fenyőfát, természetesen illő díjazásért.

A versenyt a következő feltételekkel hirdetik meg:
- A versenyen legfeljebb 50 termelő indulhat, és minden termelő legfeljebb 10 db fenyőfát nevezhet.
- A fenyőfák magasságának el kell érnie a 20 m-t, de nem haladhatja meg a 40 m-t.
- A fenyőfák átlagos átmérőjének el kell érnie a 30 cm-t, de nem haladhatja meg a 60 cm-t.
- A versenyre benevezett fenyőfák adatait – tehát a magasságot és az átlagos átmérőt – be kell küldeni a versenyszervezőkhöz.
- A versenyszervezők minden termelő esetén kiszámítják a termelő által benevezett fák összes térfogatát.
- Egy fa térfogatának kiszámításához a henger térfogatképletét alkalmazzák, amely szerint $V = r^2 \cdot \pi \cdot l$, ahol V a térfogat, r a fatörzs átlagos sugara, l pedig a fa magassága. Az egyszerűbb számítás kedvéért nem veszik figyelembe az ágakat, illetve a fa alakjának egyéb „szabálytalanságait” sem!
- Az a termelő a győztes, akinek a fái a legnagyobb összesített térfogatértéket adják az előzőekben leírt számítás alapján.
- A győztes termelő legmagasabb fája lesz a győztes fa, azaz az „Ország karácsonyfája”.

Írjon programot, amely véletlenszerűen generált adatok segítségével meghatározza a győztes termelőt, illetve a győztes fát!
- A program véletlenszerűen generálja a szükséges tesztadatokat. Ügyeljen arra, hogy a szövegben meghatározott feltételeknek megfelelő adatokat állítson elő!
- Készítsen listát, amely minden termelő esetén tartalmazza a következő adatokat:
  - Benevezett fák száma
  - Minden benevezett fa magassága, m-ben megadva, 2 tizedesjegy pontossággal
  - Minden benevezett fa átlagos átmérője, cm-ben megadva
  - A termelő fáinak összesített térfogata
- A termelőket, illetve a fákat elegendő sorszámmal azonosítani.
- A lista legyen áttekinthető, az alábbi mintának megfelelő!
- A program a leírt szabályok alapján határozza meg a győztes termelőt, illetve a győztes fát!

Minta a listázáshoz:

```
1. termelő (2 db fa)
    Sorszám      Magasság (m)    Átmérő (cm)
       1.           22,05            55
       2.           36,80            41
Összes térfogat: 10,10 m^3
```

Több termelő esetén a többi termelő adatai is hasonló módon jelenjenek meg!

A feladat megoldásaként teljes, fordítható és futtatható kódot kérünk, mely az adatokat billentyűzetről (standard input) olvassa, és a képernyőre (standard output) írja ki. Vizuális fejlesztőeszköz használata esetén az algoritmust konzol alkalmazásként (szöveges ablakban futó) kérjük elkészíteni! Beadandó: a feladatot megoldó program forráskódja!

Mintamegoldás:

- a feladat egy lehetséges, C# nyelvű megoldása
- megtalálható a *Feladat3.cs* állományban
- Az üzenetek, illetve kommentek a tördelési problémák miatt a fájlban mellékelt megoldáshoz képest néhány helyen rövidítve láthatók

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Feladat3
{
    class Karacsonyfa
    {
        private const int maxTermeloSzam = 50;  // Termelők max. száma
        private const int maxFadb = 10;  // Fák max. száma
        private int termelodb;

        private struct faadat      // Egy fa adatai
        {
            public double faMagassag;
            public int faAtmero;
        }

        private struct termeloadat  // Egy termelő fáinak az adatai
        {
            public int faDb;
            public faadat[] faAdatok;
            public double terfogat;
        }

        private termeloadat[] termeloAdatok;  // Az összes termelő adatai

        public void adatgeneralas()
        {
            Random randNum = new Random();
            termelodb = randNum.Next(maxTermeloSzam)+1; // A termelők száma
            termeloAdatok=new termeloadat[termelodb];

            Console.WriteLine(" => Adatgenerálás");
            Console.WriteLine();
            Console.WriteLine("  A termelők száma: "+termelodb);

            for (int i = 0; i < termelodb; i++)
            {
                termeloAdatok[i] = new termeloadat();
                termeloAdatok[i].faDb = randNum.Next(maxFadb) + 1;
                termeloAdatok[i].faAdatok = new faadat[termeloAdatok[i].faDb];
                termeloAdatok[i].terfogat = 0;
                Console.WriteLine();
                Console.WriteLine("  "+(i + 1) + ". termelő: (" + termeloAdatok[i].faDb+" db fa)" );
                Console.WriteLine(String.Format("{0,15}{1,15}{2,15}", "Sorszám",
                                  "Magasság (m)", "Átmérő (cm)"));
                for (int j = 0; j < termeloAdatok[i].faDb; j++)
                {
                    termeloAdatok[i].faAdatok[j] = new faadat()
                    {
                        faMagassag = (double)(randNum.Next(2001) + 2000)/100, // 20 és 40 m között
                        faAtmero = randNum.Next(31) + 30  // 30 és 60 cm között
                    };
                    Console.WriteLine(String.Format("{0,13}{1,14:0.00}{2,15}",
                                    (j + 1).ToString()+ ".",
                                    termeloAdatok[i].faAdatok[j].faMagassag,
                                    termeloAdatok[i].faAdatok[j].faAtmero));
```

```
                termeloAdatok[i].terfogat +=
                                Math.Pow(termeloAdatok[i].faAdatok[j].faAtmero*0.005, 2)
                                * Math.PI * termeloAdatok[i].faAdatok[j].faMagassag;
            }

            Console.WriteLine();
            Console.WriteLine("     Össztérfogat:
                              "+String.Format("{0,8:0.00}",termeloAdatok[i].terfogat)+" m^3");
            Console.ReadKey();
        }
    }

    private int gyoztes()
    {
        int ind = 0;
        for (int i = 1; i < termelodb; i++)
        {
           if (termeloAdatok[i].terfogat>termeloAdatok[ind].terfogat)
           {
               ind = i;
           }
        }
            return ind;
    }

    private int gyoztesfa(int termelo)
    {
        int ind = 0;
        for (int i = 1; i < termeloAdatok[termelo].faDb; i++)
        {
            if (termeloAdatok[termelo].faAdatok[i].faMagassag >
                termeloAdatok[termelo].faAdatok[ind].faMagassag)
            {
                ind = i;
            }
        }
        return ind;
    }

    public void kiiras()
    {
        int gy=gyoztes();
        Console.WriteLine();
        Console.WriteLine(" => Eredményhirdetés:");
        Console.WriteLine("      Győztes termelő sorszáma: "+ (gy+1)+".");
        Console.WriteLine("      Győztes fájának sorszáma: "+ (gyoztesfa(gy) + 1)+".");
        Console.ReadKey();
    }
}

class Program
{
    static void Main(string[] args)
    {
        Karacsonyfa k = new Karacsonyfa();
        k.adatgeneralas();
        k.kiiras();
    }
}
}
```

Értékelés:

a) A programkód szintaktikailag hibátlan, lefordítható ..... 1 pont
- Ez a pont csak abban az esetben adható meg, ha a programkód tartalmaz a b-e szakaszokba tartozó, összességében legalább 5 pontot érő részmegoldást!

b) A konstansok és változók helyes definiálása ..... 2 pont
- A termelők és fák összes adatának a tárolására alkalmas tömb, vagy tömbök definiálása: 1 pont
- Az egyéb szükséges konstansok, változók deklarálása: 1 pont

c) Adatok véletlenszerű generálása, térfogat kiszámítása ..... 5 pont
- A termelők számának, és az egyes termelők által benevezett fák számának a generálása a megadott intervallumban: 1 pont
- A fák magasságának a generálása a megadott intervallumban, 2 tizedesjegy pontossággal: 1 pont
- A fák átlagos átmérőjének a generálása a megadott intervallumban: 1 pont
- Egy-egy fa térfogatának a kiszámítása a generált adatok alapján: 1 pont
  Feltétlenül ellenőrizzük, hogy a program a számítás közben elvégzi-e az átváltást, célszerű az átmérőt cm-ről m-re váltani!
- Az összesített térfogat meghatározása minden termelő esetén: 1 pont

Megjegyzés: az előző két pont akkor is jár, ha a térfogatok a program más pontján kerülnek kiszámításra. A térfogatokat nem kötelező eltárolni!

d) Táblázatszerű kiírás: ..... 2 pont
- Minden termelő esetén megjelenik a termelő sorszáma, fáinak a száma és a fák összesített térfogata, valamint az egyes fák sorszáma, magassága és átmérője: 1 pont
- A táblázat áttekinthető, a mintának megfelelő: 1 pont

e) A győztes fa meghatározása ..... 5 pont
- Maximumkeresés tétel alkalmazása a legjobb termelő (maximális össztérfogat) meghatározására, helyes ciklusszervezés: 1 pont
- A feltétel pontos megfogalmazása: 1 pont
- Maximumkeresés tétel alkalmazása a legjobb termelő legmagasabb fájának a meghatározására, helyes ciklusszervezés: 1 pont
- A feltétel pontos megfogalmazása: 1 pont
- A győztes termelő és a győztes fa sorszámának a kiírása: 1 pont

4. feladat **15 pont**

A kitűzött feladat:

Adott az **utazas** nevű adatbázis, amely néhány utazási iroda 2010-re meghirdetett útjaival kapcsolatos adatokat tartalmaz.

Az adatbázist a vizsgabizottság által megadott helyen találhatja MS-Access 2000 formátumban. Azok számára, akik az MS-Access formátumát nem ismerő rendszerben oldják meg a feladatot, az adatbázis tábláit TXT fájlokban is megadtuk. (Az első sorban az adott tábla mezőnevei, a többi sorban az adatrekordok találhatók, a sorokon belül az adatokat pontosvessző határolja el egymástól.)

Az adatbázis elsősorban feladatkitűzési céllal készült, így nem modellezi tökéletesen a való életben felmerülő összes lehetséges helyzetet.

Az adatbázis az alábbi táblákat és relációkat tartalmazza:

irodak(
irodaazon : Egész szám -> utak.irodaazon
nev : Szöveg
szekhely : Szöveg
telefon : Szöveg
)

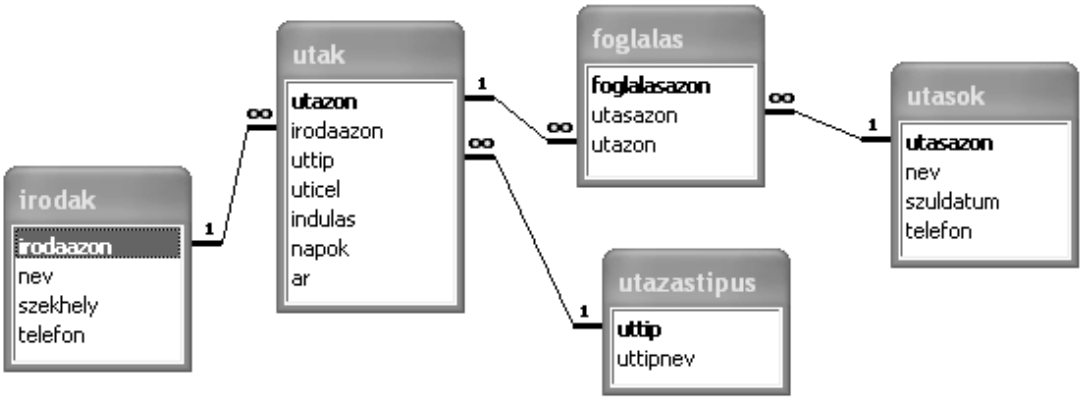
utasok(
utasazon : Egész szám -> foglalas.utasazon
nev : Szöveg
szuldatum : Dátum/Idő
telefon : Szöveg
)

utak(
utazon : Egész szám -> foglalas.utazon
irodaazon : Egész szám -> irodak.irodaazon
uttip : Egész szám -> utazastipus.uttip
uticel : Szöveg
indulas : Dátum/Idő
napok : Egész szám
ar : Pénznem
)

foglalas(
foglalasazon : Egész szám
utasazon : Egész szám -> utasok.utasazon
utazon : Egész szám -> utak.utazon
)

utazastipus(
uttip : Egész szám -> utak.uttip
uttipnev : Egész szám
)

A kettőspont után az adatmező típusát adtuk meg, a „->” karakterek után pedig a más táblákkal való kapcsolatot. Az elsődleges kulcsot aláhúzás jelöli.



Az **irodak** adattábla tartalmazza az utazási irodák egyedi azonosítóját, nevét, székhelyét, és telefonszámát.

Az **utasok** adattábla tartalmazza azoknak az utasoknak a személyes adatait, akik korábban már utaztak valamelyik utazási irodával. Az adatok a következők: egyedi utas azonosító, név, születési dátum és telefonszám.

Az **utak** adattábla tartalmazza az utazási irodák által 2010-re meghirdetett utak adatait: egyedi út azonosító, az utat szervező iroda azonosítója, az utazás típusának a kódja (lehetséges értékek: 1-egyéni, 2-buszos, 3-repülő, a megfeleltetéseket az **utazastipus** adattábla írja le), úti cél, indulás ideje, az út napokban megadott időtartama, az út ára.

A **foglalas** adattábla tartalmazza azt, hogy ki, milyen utat foglalt már magának 2010-re. A foglalások adatai: egyedi foglalás azonosító, az utas azonosítója, a lefoglalt út azonosítója.

Az **utazastipus** adattábla tartalmazza, hogy az egyes utazástípus kódok milyen utazástípust jelentenek.

A. Készítsen lekérdezést, amely megadja az összes olyan foglalást, amely repülős útra vonatkozik, és az út ára 100000 és 200000 Ft közé esik (beleértve a határokat is)!
A lekérdezés eredményében szerepeljen az utas neve, az úti cél, valamint az indulás és érkezés ideje! A számított mező neve legyen *erkezes*! A lista legyen az utasok neve szerint növekvően rendezett!

B. Készítsen lekérdezést, amely megadja, hogy az egyes utazási irodáknál (*iroda*) hányan foglaltak már utazást 2010-re (*utasszam*), és ebből eddig milyen bevétele származott az egyes irodáknak (*bevetel*), ha az utasoknak minden irodában a részvételi díj 30%-át kellett előlegként befizetni! A lista elsődlegesen az utasok száma szerint csökkenően, másodsorban a bevétel szerint növekvően legyen rendezett!
Az eredménytábla oszlopnevei a zárójelben megadott nevek legyenek!

C. Készítsen lekérdezést, amely megadja az adatbázisban nyilvántartott utasok közül az olyanok számát, akik még nem foglaltak semmilyen utazást 2010-re!
Az eredménytábla oszlopneve legyen *nemutazok*!

Megoldás, értékelés:

a) Lekérdezés A ..... 5 pont
- A lista a megadott mezőket tartalmazza[1]:1 pont
- A számított mező képlete és elnevezése helyes[2] : 1 pont
- Az árra és utazástípusra vonatkozó szűrőfeltétel helyes (az ár esetében a Between operátor is alkalmazható)[3]: 1 pont
- A táblák közötti kapcsolat helyes[4]: 1 pont
- A lista az utasok neve szerint növekvően rendezett[5]:1 pont

Egy lehetséges megoldás:

```
SELECT utasok.nev, uticel, indulas[1], indulas+napok-1 AS erkezes[2]
FROM foglalas, utak, utasok, irodak, utazastipus[1]
WHERE ((ar>=100000 And ar<=200000) AND
            (utazastipus.uttipnev="repülős")[3] AND
            (utasok.utasazon=foglalas.utasazon) AND
            (irodak.irodaazon=utak.irodaazon) AND
            (utak.utazon=foglalas.utazon) AND
            (utak.uttip=utazastipus.uttip))[4]
ORDER BY utasok.nev[5];
```

b) Lekérdezés B ..... 5 pont
- A lista az irodák szerint csoportosított[1]:1 pont
- A lista megadja az irodák nevét, valamint irodánként az utasok számát a megfelelő oszlopnévvel[2]: 1 pont
- A lista megadja irodánként az utasok által befizetett összes előleget a megfelelő oszlopnévvel[3]: 1 pont
- A táblák közötti kapcsolat helyes[4]: 1 pont
- A lista elsődlegesen az utasok száma szerint csökkenően, másodsorban a bevétel szerint növekvően rendezett[5]:1 pont

```
SELECT irodak.nev AS iroda, Count(utasok.utasazon) AS utasszam[2],
            Sum(ar)*0.3 AS bevetel[3]
FROM foglalas, utak, utasok, irodak
WHERE ((utasok.utasazon=foglalas.utasazon) AND
            (irodak.irodaazon=utak.irodaazon) AND
            (utak.utazon=foglalas.utazon))[4]
GROUP BY irodak.nev[1]
ORDER BY Count(utasok.utasazon) DESC , Sum(ar)*0.3[5];
```

c) Lekérdezés C ..... 5 pont
- A lekérdezés beágyazott lekérdezést alkalmaz[1]: 1pont
- A beágyazott lekérdezés megadja az összes utas azonosítóját, aki foglalt utat[2]: 1 pont
- A Not In operátor helyes alkalmazása, helyes szűrőfeltétel[3]: 1 pont
- A nem utazók számának az összesítése[4]: 1 pont
- Az oszlopnév megjelenítése[5]: 1 pont

```
SELECT Count(utasok.utasazon)[4] AS nemutazok[5]
FROM utasok
WHERE ((utasok.utasazon Not In[3]
            (SELECT utasazon FROM foglalas)[1,2]));
```