

# **BUDAPESTI KOMPLEX SZAKKÉPZÉSI CENTRUM WEISS MANFRÉD TECHNIKUM, SZAKKÉPZŐ ISKOLA ÉS KOLLÉGIUM**

**Szakképesítés azonosító száma, megnevezése**  
**5 0613 12 03 Szoftverfejlesztő és -tesztelő**

**Vizsgatevékenység:**  
**Projektfeladat**

## **Asztali és webes szoftverfejlesztés, adatbázis-készítés**

### **Minta feladatsor**

**A vizsgafeladat időtartama: 240 perc**

**Elérhető pontszám: 65 pont**

**Értékelés:** a projektfeladat eredménye a vizsgaremek értékelésére kapott pontszámokkal összevontan kerül osztályozásra.

**Készítették:**

Fodor Péter  
Kovács László Bálint

**2022**

## 1. Asztali alkalmazás fejlesztés

### Kalapácsvetés – konzolos feladat

A kalapácsvetés az atlétika egyik versenyszáma. Minden idők legjobb férfi kalapácsvetőinek rangsorát a *kalapacsvetes.txt* szöveges állomány tartalmazza, a 2021. júniusi állapotnak megfelelően. A fájl UTF-8 kódolású és minden sorában egy sportoló adatai szerepelnek az alábbi sorrendben:

- helyezés,
- eredmény (méterben, két tizedesjegy pontossággal)
- sportoló neve,
- a sportoló országkódja,
- helyszín,
- dátum.

Az állomány első sora a **mezőneveket** tartalmazza, az adatokat **pontosvessző** választja el a mintának megfelelően:

Helyezés	Eredmény	Sportoló	Országkód	Helyszín	Dátum
1	86,74	Yuriy Sedykh	URS	Stuttgart	1986.08.30
2	86,04	Szergej Litvinov	URS	Drezda	1986.07.03
3	84,90	Vadim Devyatovskiy	BLR	Minszk	2005.07.21
4	84,86	Koji Murofushi	JPN	Prága	2003.06.29

Megoldását úgy készítse el, hogy az azonos szerkezetű, de tetszőleges bemeneti adatok esetén is helyes eredményt adjon!

A képernyőre írja ki a feladat sorszámát az eredmény megjelenítése előtt!

A feladatokban a kiírásokat a minta szerint készítse el!

### Feladatok

1. Készítsen **konzolos** alkalmazást és mentse el a projektet *Kalapacsvetes* néven! (1 pont)
2. Hozzon létre saját osztályt *Sportolo* azonosítóval, melynek adatait felhasználva egy-egy sportoló adatait tudja majd tárolni! Készítse el a *Sportolo* osztály konstruktorát, mely hívásával az osztály adatait tudja inicializálni! (2 pont)
3. Olvassa be a *kalapacsvetes.txt* állomány sorait és tárolja az adatokat egy *Sportolo* osztályon alapuló összetett adatszerkezetben! Ügyeljen arra, hogy az állomány első sora az adatok fejlécét tartalmazza! (2 pont)
4. Határozza meg és írja ki, hány dobás eredménye található a forrásfájlban! (1 pont)
5. Határozza meg és jelenítse meg a forrásállományban szereplő magyar (HUN) sportolók dobásainak átlageredményét! Az eredményt két tizedesre kerekítve írja ki! (2 pont)
6. Kérjen be egy évszámot és írja ki, hogy abban az évben mennyi dobás került be a legjobbak közé, illetve írja ki, hogy mely sportolók érték el ezeket. Ellenkező esetben írja ki, hogy az adott évben nem került be egy dobás eredménye sem a legjobbak közé. (3 pont)

7. Készítsen statisztikát, hogy melyik országból hány kalapácsvetés eredménye szerepel a legjobb dobások között. Az eredményt a mintának megfelelően írassa ki a képernyőre! (2 pont)
8. Hozzon létre *magyarok.txt* néven egy UTF-8 kódolású fájlt, amelyben csak a magyar (HUN)sportolók eredményei szerepelnek. (2 pont)

Képernyőkép:

```
4. feladat: 25 dobás eredménye található.
5. feladat: A magyar sportolók átlagosan 83,39 métert dobtak.
6. feladat: Adjon meg egy évszámot:
1998
    3 darab dobás került be ebben az évben.
    Tibor Gécsek
    Balázs Kiss
    Szergej Kirmaszov
7. feladat: Statisztika
    URS - 5 dobás
    BLR - 3 dobás
    JPN - 1 dobás
    HUN - 4 dobás
    POL - 2 dobás
    RUS - 2 dobás
    RPG - 1 dobás
    FIN - 1 dobás
    DEU - 2 dobás
    USA - 1 dobás
    NDK - 1 dobás
    UKR - 1 dobás
    SVN - 1 dobás
```

Képernyőkép, ha a megadott évben nem került be egy dobás sem:

```
4. feladat: 25 dobás eredménye található.
5. feladat: A magyar sportolók átlagosan 83,39 métert dobtak.
6. feladat: Adjon meg egy évszámot:
2010
    Egy dobás sem került be ebben az évben.
7. feladat: Statisztika
    URS - 5 dobás
    BLR - 3 dobás
    JPN - 1 dobás
    HUN - 4 dobás
    POL - 2 dobás
    RUS - 2 dobás
    RPG - 1 dobás
    FIN - 1 dobás
    DEU - 2 dobás
    USA - 1 dobás
    NDK - 1 dobás
    UKR - 1 dobás
    SVN - 1 dobás
```

## Függőhidak – formos feladat

Készítsen grafikus, két formból álló, menüvezérelt alkalmazást a mintának megfelelően a függőhidak adatainak megjelenítésére és kezelésére. A legnagyobb támaszközü függőhidak listáját a *fuggohidak.csv* nevű, UTF-8 kódolású fájl tartalmazza.

A forrásfájl sorai egy-egy híd adatait tartalmazzák a következő sorrendben:

- helyezés, hosszúság szerint csökkenő sorrendben,
- a híd neve,
- földrajzi hely, ahol a híd található,
- ország,
- a híd hossza (támaszközök közti távolsága) méterben,
- átadás éve.

Az állomány első sora a **mezőneveket** tartalmazza, az adatokat **tabulátor** választja el egymástól.

Függőhidak

Keresés    Kilépés

1915 Çanakkale híd  
Akasi Kaikjő híd  
Xihoumen-híd  
Nagy-Bélt híd  
Osman Gazi híd  
I Szunsin híd  
Runyang-híd  
Negyedik nankingi Jangce-híd  
Humber híd  
Vitéz Szelim szultán híd  
Jiangyin-híd  
Tsing-Ma-híd  
Hardanger híd  
Verrazano-Narrows híd

Hely: Zhoushan

Ország: Kína

Hossz: 1650

Év: 2008

Hidak száma

☒ 2000 előtt épült  
☐ 2000-ben vagy utána

26 darab

Kilépés

Keresés

Akasi Kaikjő híd  
Déli Bisan-Seto-híd  
Harmadik Kurushima Kaikyo híd  
Második Kurushima Kaikyo híd

Keresés

Ország: Japán

☒ 1 km-nél hosszabbak

Keresés    Bezárás

## Feladatok

1. Hozzon létre saját osztályt *Fuggohid* azonosítóval, melynek adattagjait felhasználva egy-egy híd adatait tudja majd tárolni és készítse el a *Fuggohid* osztály konstruktorát, mely hívásával az osztály adattagjait tudja inicializálni! A projektet *Fuggohidak* néven mentse el! (2 pont)
2. A Fájl/Megnyitás menüpontot kiválasztva jelenjen meg a megnyitás párbeszéd ablak és olvassa be a *fuggohidak.csv* állomány sorait és tárolja az adatokat egy *Fuggohid* osztályon alapuló összetett adatszerkezetben! (2 pont)
3. A függőhidak nevei egy ListBox-ban jelenjenek meg, a függőhidak további adatai szövegdobozokban jelenjenek meg, amikor kiválasztunk egy hidat a listából. (1 pont)
4. Rádiógombok segítségével jelenítse meg a 2000 előtt, illetve 2000-ben és utána épült hidak számát! (1 pont)

5. Hozza létre a főmenüben a Keresés és a Kilépés menüpontokat, a Kilépés menüpont és a Kilépés gomb hatására záródjon be az alkalmazás! (1 pont)
6. A Keresés menüpont hatására jelenjen meg a keresés form, a főformot rejtse el, csak a bezárás gombra kattintva jelenjen meg újra! (1 pont)
7. A keresés form megjelenésekor a legördülő lista (ComboBox) elemeit dinamikusan töltse fel a forrásfájlban szereplő országok neveivel! (1 pont)
8. A Keresés gomb hatására a megadott feltételeknek megfelelő függőhidak nevei jelenjenek meg a szövegdobozban. (1 pont)



Adatok forrása:

[https://hu.wikipedia.org/wiki/A\\_legnagyobb\\_t%C3%A1maszk%C3%B6z%C5%B1\\_f%C3%BCgg%C5%91hidak\\_list%C3%A1ja](https://hu.wikipedia.org/wiki/A_legnagyobb_t%C3%A1maszk%C3%B6z%C5%B1_f%C3%BCgg%C5%91hidak_list%C3%A1ja)

## 2. Komplex webes és adatbázis-kezelési feladat

### Névnepkereső

Egyes weboldalakon gyakran előforduló tartalom, hogy feltüntetik az aktuális névnapokat, vagy akár az elkövetkező napokon ünneplők neveit is. Ehhez a háttérrel például egy REST API biztosíthatja, melynek bemenete egy dátum, eredménye pedig a dátumhoz tartozó névnap, vagy névnapok. Megfordítva a funkciót, akár arra is szükség lehet, hogy egy felületen a látogatók egy-egy névnaphoz tartozó dátumot tudjanak keresni.

Az ön feladatai most egy a fentiekben leírtaknak megfelelő REST API-ra vonatkoznak, melyhez az alábbi adatforrást kell felhasználnia:

- <https://infojegyzet.hu/forrasfajlok/nevnapok.sql>

Önnek a folytatásban készítenie kell egy API-t, és ezen API felhasználásával egy reszponzív viselkedésű webes felületet a következők szerint!

a.) A REST API elkészítésére vonatkozóan:

1. A leendő API működésének előkészítéseként importálja a `nevnapok.sql` fájlból az adatbázist a saját gépére a localhost phpMyAdmin felületén keresztül.
2. Egészítse ki az adatbázis egyetlen tábláját úgy, hogy az rendelkezzen egy új mezővel:
  - az új mezőt a tábla jelenlegi első eleme elé szúrja be,
  - szerepe szerint kulcsmező legyen, automatikusan növekvő sorszámozással.
3. Hozzon létre API végpontot, mely az alábbi címen legyen elérhető:

- <http://localhost/api/nevnapok/>

4. A végpontot úgy alakítsa ki, hogy az paraméterezhető legyen egy dátummal. Ez a dátum – a lentebb látható módon pl. április 30. napját jelölve – csak hónapot és napot kell, hogy tartalmazzon (kötőjellel elválasztva), évszámot nem:

- <http://localhost/api/nevnapok/?nap=4-30>

5. Fenti hívás esetén az API által visszaadott JSON kód a következők szerint alakuljon:

```
{"datum": "április 30.", "nevnap1": "Katalin", "nevnap2": "Kitti"}
```

6. Bővítse az API paraméterezési lehetőségét névkeresési funkcióval is az alábbi hívási minta szerint:

- <http://localhost/api/nevnapok/?nev=Katalin>

Találat esetén az API által visszaadott JSON kód ebben az esetben is az 5. pontnak megfelelő szerkezetű legyen. (Amennyiben a keresett névhez több dátum is tartozna, ezúttal most elegendő csupán egy találatot visszaadni.)

7. Az API hívása paraméter hiányában adjon JSON formátumú útmutatást arra vonatkozóan, hogyan kell használni az API-t:

```
{"minta1":"/?nap=12-31","minta2":"/?nev=Szilveszter"}
```

8. Az API találat hiányában, vagy rossz paraméter esetén szintén JSON formában adjon hibaüzenetet:

```
{"hiba":"nincs találat"}
```

b.) A webes felületre és a REST API használatára vonatkozóan:

9. Hozzon létre űrlapot, ahol névnapkeresés céljából a felhasználónak lehetősége van megadnia

- vagy az év valamely napját (évszám nélkül, hónappal, és nappal),
- vagy pedig egy keresett keresztnévet.

Mindehhez tetszőleges űrlapelem(ek)et, vagy bármilyen más webes elemeket használhat.

10. Az űrlapon helyezzen el keresést kezdeményező párbeszédelemet (gombot) is. Ennek módja szintén önre van bízva: megoldhatja a két külön keresési feladatot közös gombbal, és külön-külön gombokkal is.

11. A webes felület kialakítása során gondoskodjon róla, hogy a leendő találati eredményeknek is legyen majd helye az oldalon anélkül, hogy a keresőfelület eltűnne onnan.

12. Gondoskodjon arról is, hogy az elkészített felület asztali számítógépen (széles képernyőn), és hordozható eszközön (álló tájolású kijelzőn) egyaránt élvezhető, felhasználóbarát élményt nyújtson. Ennek keretében:

- széles képernyőn inkább egymás mellett, álló tájolás esetén inkább egymás alatt jelenítsen meg tartalmakat,
- széles képernyőn nagyobb margókat, álló tájolás esetén kisebb margókat állítson be,
- mindkét esetben ügyeljen a szövegeknél az optimális, jól olvasható betűméretre.

13. Oldja meg, hogy a keresés indításának hatására a program az ön által az a.) feladat során elkészített API használatával keressen az adatbázisban. Az API-t az űrlap adataival paraméterezve hívja meg.

Amennyiben az a.) feladatban a REST API-t nem tudja teljesen, vagy részben elkészíteni, úgy a b.) feladat során a saját változata helyett az alábbi API-t is használhatja:

- <https://infojegyzet.hu/apiminta/nevnapok/>

14. Az API által visszaadott találati eredményeket (vagy az annak eredménytelenségére utaló választ) jelenítse meg a felületnek a 11. pontban erre a célra kialakított részén. Ennek értelmében:

- dátum szerinti keresés esetén írassa ki, hogy a keresett napon ki, vagy kik ünneplik a névnapjukat,
- keresztnév szerinti keresés esetén írassa ki az API által visszaadott dátumot,
- eredménytelen keresés esetén adjon erre utaló szöveges információt,
- üresen hagyott űrlapról történő keresés esetén pedig adjon rövid szöveges utasítást a felhasználó számára az űrlap használatára vonatkozóan.

Elkészült feladatát mutassa be szóban vizsgáztatójának!



A vizsgázó neve: .....

## Értékelő lap

### 1. Asztali alkalmazás fejlesztés

*A konzolos feladat vonatkozásában (15 pont):*

1. Létrehozta a konzolos alkalmazást és elmentette a projektet *Kalapacsvetes* néven ..... 1 pont
2. Létrehozta a saját osztályt *Sportolo* azonosítóval és az osztály adattagjai alkalmasak a feladat adatainak tárolására ..... 1 pont  
Elkészítette az osztály konstruktorát, melyben az adattagokat inicializálja..... 1 pont
3. Beolvasta a *kalapacsvetes.txt* állomány sorait ..... 1 pont  
Tárolta az adatokat egy *Sportolo* osztályon alapuló összetett adatszerkezetben. 1 pont
4. Kiírja a látogatások számát a képernyőre..... 1 pont
5. Kiszámolja az átlagot ..... 1 pont  
Az eredményt két tizedesjegyre kerekítve jeleníti meg ..... 1 pont
6. Bekéri az IP-címet és tárolja egy változóban ..... 1 pont  
Kiírja a megfelelő dátumokat ..... 1 pont  
Kiírja, ha nem volt látogatás a megadott IP-címről ..... 1 pont
7. Elkészítette a statisztikát ..... 1 pont  
Az eredményt a mintának megfelelően kiíratta a képernyőre..... 1 pont
8. Létrehozta *magyarok.txt* UTF-8 kódolású fájlt ..... 1 pont  
A fájlban a feladatnak megfelelő sorok szerepelnek..... 1 pont

*A formos feladat vonatkozásában (10 pont):*

1. Létrehozta a saját osztályt *Fuggohid* azonosítóval és az osztály adattagjai alkalmasak a feladat adatainak tárolására. .... 1 pont  
Elkészítette az osztály konstruktorát, melyben az adattagokat inicializálja..... 1 pont

2. Beolvasta a *fuggohidak.csv* állomány sorait ..... 1 pont  
Tárolja az adatokat egy *Fuggohid* osztályon alapuló összetett adatszerkezet-  
ben ..... 1 pont
3. A cégek nevei egy ListBox-ban, a további adatok szövegdobozokban  
megjelennek ..... 1 pont
4. Rádiógombok segítségével megjeleníti a feltételeknek megfelelő cégek  
számát..... 1 pont
5. Létrehozta a menüpontokat és működik a kilépés ..... 1 pont
6. Létrehozta a második formot és az megfelelően megjelenik a menüpont  
hatására, illetve bezárás után az első form megjelenik ..... 1 pont
7. A legördülő lista elemeit dinamikusan feltöltötte az országok neveivel..... 1 pont
8. A Keresés gomb a feladatnak megfelelően működik..... 1 pont

## 2. Komplex webes és adatbázis-kezelési feladat

*A backend programozás vonatkozásában (15 pont):*

9. Az adatbázist importálta..... 1 pont
10. Létrehozta az új mezőt ..... 1 pont  
Az új mezőt kulcsmezőnek állította be..... 1 pont  
Az új mező automatikusan növekvő számozású ..... 1 pont
11. A végponthoz létrehozta a programot ..... 1 pont
12. A program képes fogadni a dátum-paramétert ..... 1 pont  
A program megfelelően feldolgozza a paramétert ..... 1 pont
13. Az adatbázis-kapcsolatot felépítette ..... 1 pont  
Az adatbázisban helyesen keres ..... 1 pont  
Az eredményt JSON kódban visszaadja..... 1 pont
14. A program képes fogadni a név-paramétert ..... 1 pont  
A program megfelelően feldolgozza a paramétert ..... 1 pont  
Az adatbázisban helyesen keres ..... 1 pont
15. A paraméter hiányát helyesen kezeli..... 1 pont
16. Az eredménytelen keresést megfelelően kezeli ..... 1 pont

*A rezponzív viselkedésű weboldal vonatkozásában (10 pont):*

- 17. A dátum megadását lehetővé tette..... 1 pont
  - A dátum bekérése megfelelő formátumban történik ..... 1 pont
  - A keresztnév megadását lehetővé tette..... 1 pont
  - A keresztnév bekérése megfelelő párbeszédelemmel történik..... 1 pont
- 18. Létrehozott keresés indítására párbeszédelemet (gombot)..... 1 pont
  - A párbeszédelemhez (vagy a formhoz) társított programot ..... 1 pont
- 19. Kialakította a helyet a leendő találati eredménynek..... 1 pont
- 20. Képernyőtájolás függvényében helyezi el a tartalmakat..... 1 pont
  - Képernyőtájolás függvényében állított be margót ..... 1 pont
  - Képernyőtájolás függvényében állított be betűméretet ..... 1 pont

*A frontend programozás vonatkozásában (15 pont):*

- 21. Az API-t helyesen hívta a dátummal..... 2 pont
  - Az API-t helyesen hívta a keresztnévvel..... 2 pont
  - A kapott keresztnéve(ke)t helyesen olvassa ki a JSON kódból ..... 1 pont
  - A kapott dátumot helyesen olvassa ki a JSON kódból..... 2 pont
- 22. A kapott keresztnéve(ke)t helyesen írja képernyőre ..... 2 pont
  - A kapott dátumot helyesen írja képernyőre..... 2 pont
  - Az eredménytelen keresést jelzi a képernyőn ..... 1 pont
  - A paraméter nélküli API hívásról visszajelzést ad..... 1 pont

Elért összes pontszám (max. 65 pont): ..... pont

***Budapest, 2022. május 19.***

.....  
aláírás