Szkriptnyelvek - JavaScript ismertető

- A programot JavaScript nyelven kell megírni.
- A benyújtandó fájl neve: `feladat.js`
    - Egy JavaScript nyelven írt, szöveges fájl (nem zip, rar, stb.)
    - Ez csak a feladatban kért dolgokat tartalmazza! Amennyiben saját inputtal teszteled a kódot lokálisan, úgy feltöltés előtt a tesztelő kódrészletet kommenteld ki!
- A megoldást Bíró2 webes felületén (https://biro2.inf.u-szeged.hu) keresztül kell benyújtani és a megoldást a Bíró fogja kiértékelni.
    - A Feladat beadása felületen a *Feltöltés* gomb megnyomása után ki kell várni, amíg lefut a kiértékelés. **Kiértékelés közben nem szabad az oldalt frissíteni vagy a Feltöltés gombot újból megnyomni** különben feltöltési lehetőség veszik el!
- Feltöltés után a Bíró a programot **Node** interpreterrel fogja futtatni, és különböző tesztesetekre futtatja.
- A program működése akkor helyes, ha a tesztesetek futása nem tart tovább 2 másodpercnél és hiba nélkül fejeződik be, valamint a program működése a feladatkiírásnak megfelelő.
- Ha 3 teszteset futási ideje túllépi a fenti időkorlátot, a tesztelés befejeződik, a pontszám az addig szerzett pontszám lesz.
- A riport.txt megtekinthető az alábbi módon:
    1. Az Eredmények megtekintése felületen a vizsgálandó próba új lapon való megnyitása
    2. A kapott url formátuma: https://biro2.inf.u-szeged.hu/Hallg/IB370G/FELADAT_SZAMA/hXXXXXX/4/riport.txt
    3. Az url-ből visszatörölve a 4-esig (riport.txt törlése) megkaphatók a 4-es próbálkozás adatai
- A programot 25 alkalommal lehet benyújtani, a megadott határidőig.
- A munkád során figyelj arra, hogy pontosan kövesd a feladatban leírtakat, az elnevezéseket!
- A fájl elejére kommentbe írd be a neved, Neptun és h-s azonosítód az alábbi formában:

```
# Nev: Vezeteknev Keresztnev
# Neptun: NEP4LF
# h: h123456
```

Tökéletes szám

Készíts egy `tokeletes` nevű függvényt. A függvény egy számot vár paraméterként (ezt ellenőrizd le!). A számelméletben tökéletes számnak nevezzük azokat a természetes számokat, amelyek megegyeznek az önmaguknál kisebb osztóik összegével. A függvény a kapott számról döntse el, hogy tökéletes-e vagy sem. Ha a szám tökéletes, térjen vissza igaz értékkel, különben hamissal.

Példa

```
tokeletes(6) // true
tokeletes(28) // true
tokeletes(16) // false
tokeletes("szoveg") // false
```

Webböngésző

- Készíts egy olyan `webbongeszo` nevű osztályt, amely egy webböngészőt reprezentál. Három adattagja legyen: a nyitott lapok (*tömb*, kezdetben üres, a neve legyen **lapok**), a rendelkezésre álló memória(*szám*), valamint a memóriafogyasztás (*szám*).
- Legyen egy konstruktora, amely várja a rendelkezésre álló memóriát, ha nem adjuk meg ezt az adatot, legyen 8192 Mb.
- Készíts a memóriafogyasztás szabályozására get és set propertyket, "memoriafogyasztas" néven. A memóriafogyasztás nem lehet nullánál kisebb, sem pedig nagyobb, mint a rendelkezésre álló memória. Az intervallumon kívül eső érték esetén a megfelelő határértéket vegye fel a fogyasztás. Tehát például ha a rendelkezésre álló memória 1000 és a settert 1500 értékkel hívják meg, akkor legyen az új érték 1000.
- Készíts egy `ujLap` metódust, ami a megnyitott lapokhoz hozzáadja a paraméterben érkező URL-t. Ez a lap növelje a memóriafogyasztást véletlenszerűen 100-1500 Mb-tal. Véletlen szám generálásához használd az alábbi kódot: `Math.floor(Math.random() * 11);`. Ez egy véletlenszerű egész számot ad vissza 0 és 10 között (0 is és 10 is lehet).
- Készíts egy `lapBezar` metódust. Ez a metódus nem vár paramétert, a legutoljára hozzáadott megnyitott URL-t tartalmazó lapot zárja be, majd csökkenti a memóriafogyasztást véletlenszerűen 30-1000 Mb-tal.
- Készíts egy `panik` metódust. Ez ürítse ki a megnyitott lapokat tartalmazó tömböt, majd mérsékelje a memóriafogyasztást 10 Mb-ra.

Jó munkát!