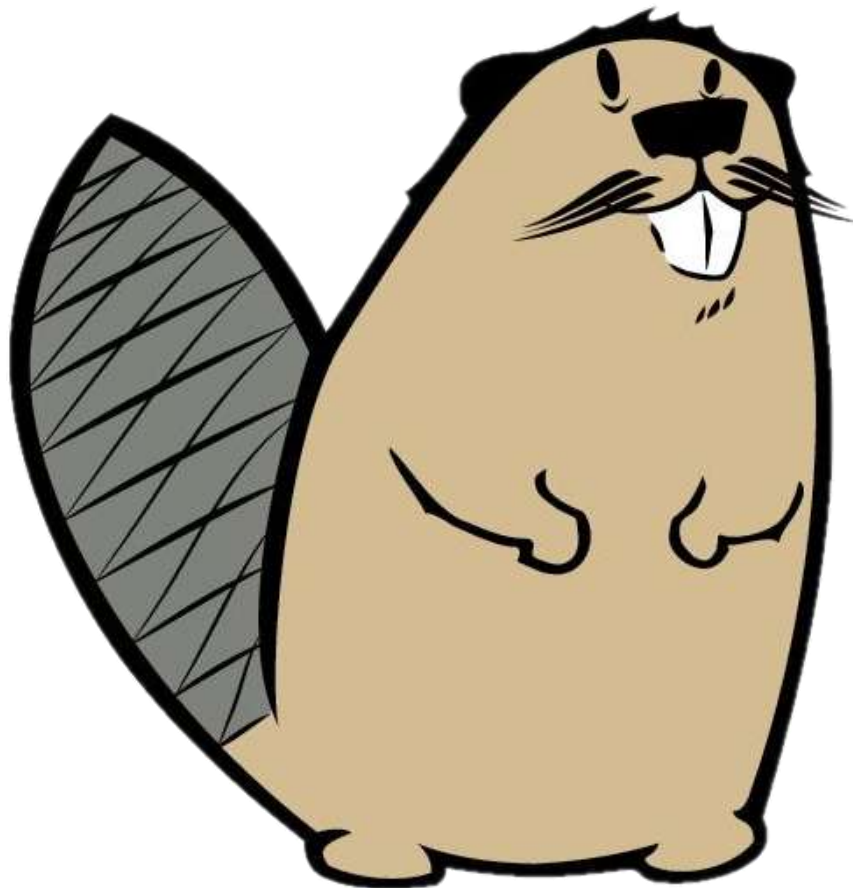


# HÓDÍTSD MEG A BITEKET!

INFORMATIKAI GONDOLKODÁST TÁMOGATÓ, NEMZETKÖZI BEBRAS  
KEZDEMÉNYEZÉS MAGYAR MEGVALÓSULÁSA



2021

## MI IS AZ E-HÓD

Az e-HÓD/HÓDítsd meg a biteket a nemzetközi BEBRAS-kezdeményezés magyar partnere.

A nemzetközi Bebras, melyhez 2020-ban már 50 ország kapcsolódott, 2015-ben elnyerte az Informatics Europe „Best Practices in Education” díját.

A kezdeményezés alapja Dr. Valentina Dagiene litván professzor által életre keltett verseny, melynek célja, hogy rövid, gyorsan (kb. 3 perc alatt) megérthető és megoldható feladatokkal megvalósítsa az alábbiakat:

- felkeltse az érdeklődést az informatika iránt;
- feloldja az informatikával kapcsolatos féltelmeket, negatív érzéseket;
- megmutassa az informatika sokszínűségét, felhasználási lehetőségeit és területeit.

A kérdések három nehézségi szinten csak strukturált és logikus gondolkodást igényelnek, semmilyen különleges informatikai tudás nem szükséges a megválaszolásukhoz. A feladatok érdekes problémákat mutatnak be. Nem tesztek, inkább szórakoztató gondolkodtató feladványok.

Magyarországon 2021-ben tizenegyedik alkalommal, öt korcsoportban vehettek részt a diákok 4-től 12. osztályig.

A versenyt az ELTE IK és az NJSZT Közoktatási Szakosztálya szervezi.

Az alábbi dokumentumban a 2021-es magyar verseny feladatai és megoldásai találhatóak.

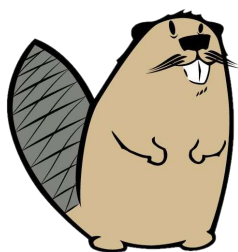
További információkért látogasson el a [HTTP://E-HOD.ELTE.HU/](http://E-HOD.ELTE.HU/) weboldalra, vagy írjon email-t az [info@e-hod.elte.hu](mailto:info@e-hod.elte.hu) címre.

## RÉSZVÉTEL

A részvétel mindenki számára ingyenes.

A verseny november második és harmadik hetében kerül lebonyolításra, osztályonként kiválasztható, hogy az adott héten melyik napon mikor oldják meg a feladatokat (8:00 és 14:00 között). Ezzel biztosítható, hogy akár egy tanóra keretein belül tudjanak részt venni egész osztályok.

A résztvevő diákoknak egy-egy internet kapcsolattal rendelkező számítógépre van szükségük. A feladatok megjelenítése és elküldése minden böngészőn működik. A verseny befejezése után, a hód hetet követően kerülnek nyilvánosságra a megoldások, melyek lehetőség szerint átbeszélhetők ugyancsak akár egy tanóra keretein belül.



## SZABÁLYOK

- A verseny lebonyolítása iskolai helyszíneken történik.
- A résztvevők online kapják meg és válaszolják meg a kérdéseket;
- A versenyre fordítandó idő 45 perc, 18 feladat három nehézségi szinten: könnyű, közepes és nehéz (legkisebb korosztályban 12 feladat);
- A verseny alatt semmilyen más számítógépes program, alkalmazás nem használható;
- A verseny során nyugalmas környezetet kell biztosítani;
- A terem a verseny során nem hagyható el;
- Az esetleges számítógéppel, internettel kapcsolatos észrevételeket a kontakt személynek kell összegyűjtenie és továbbítani a szervezők felé;
- A verseny célja: minél több pont összegyűjtése helyes válaszok megjelölésével, helytelen válaszok esetén pontlevonás történik;
- A kérdések tetszőleges sorrendben megválaszolhatók;
- A kérdések, problémák megértése a feladat részét képezi. Ezért a feladatok megbeszélése és értelmezéssel kapcsolatos kérdések nem megengedettek;
- A megoldások a verseny befejezése után, a hód hetet követően kerülnek nyilvánosságra.

## ÉRTÉKELÉS, PONTOZÁS

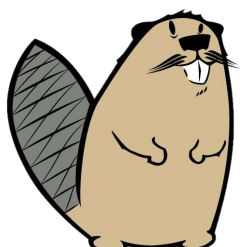
A Kishód korcsoportban 12, minden más korcsoportban 18 feladatot kell megoldani három nehézségi szinten. Minden helyes válasz pontot ér, minden helytelen válaszáért pontlevonás jár.

Nem megválaszolt kérdés esetében az összpontszám változatlan marad.

Az alábbi táblázat mutatja, hogy a feladatok nehézségétől függően hány pont kerül jóváírásra, illetve levonásra:

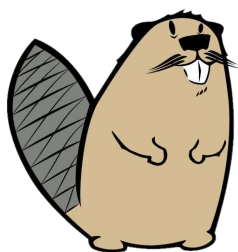
|                  | Könnyű  | Közepes | Nehéz   |
|------------------|---------|---------|---------|
| Helyes válasz    | 6 pont  | 9 pont  | 12 pont |
| Helytelen válasz | -2 pont | -3 pont | -4 pont |

Összesen (18 feladat esetében) maximum 216 pont érhető el, mivel mindenki 54pontról indul.

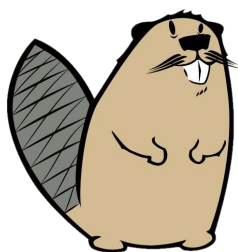


## TARTALOMJEGYZÉK

|                                       |    |
|---------------------------------------|----|
| Mi is az E-HÓD.....                   | 2  |
| Részvétel.....                        | 2  |
| Szabályok.....                        | 3  |
| Értékelés, Pontozás.....              | 3  |
| Tartalomjegyzék.....                  | 4  |
| Feladatok.....                        | 6  |
| Bináris Kapuk (2017-AZ-02).....       | 7  |
| Két hód dolgozik (2018-LT-04).....    | 9  |
| Megfigyelés (2021-AT-01).....         | 12 |
| Truchet (2021-AT-06).....             | 15 |
| Központi kórházak (2021-BE-02).....   | 18 |
| Pötyik (2021-CA-01b).....             | 20 |
| Pókháló (2021-CA-02).....             | 23 |
| Epertolvaj (2021-CH-04c1).....        | 25 |
| Tömör megjelenítés (2021-CH-06).....  | 27 |
| Sapka, korong (2021-CH-07a).....      | 30 |
| Gyümölcs tízórára (2021-CH-13).....   | 33 |
| Magasságkorlátozás (2021-CY-03).....  | 35 |
| Önvezető padlórobot (2021-CZ-04)..... | 37 |
| Ellenőrző Bizottság (2021-CZ-05)..... | 42 |
| Érmepiramis (2021-CZ-06).....         | 46 |
| Robotkar (2021-DE-03).....            | 48 |
| Bólintás számlálása (2021-DE-05)..... | 50 |
| A teknős útja (2021-DE-07).....       | 53 |
| Kedvenc ajándék_a (2021-DE-08a).....  | 56 |
| Kedvenc ajándék_b (2021-DE-08b).....  | 58 |
| Viharkár (2021-HU-02).....            | 61 |
| Felhasználónév (2021-HU-03).....      | 64 |

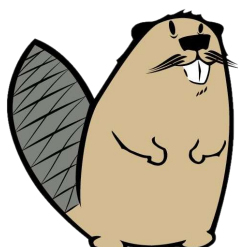


|                                           |     |
|-------------------------------------------|-----|
| Hódrally (2021-HU-04) .....               | 66  |
| Nyomda (2021-HU-05c) .....                | 69  |
| Színes folyadékok (2021-ID-10) .....      | 71  |
| Mez (2021-IE-04) .....                    | 73  |
| Formabontó (2021-IN-05) .....             | 75  |
| Zsetonoszlopok (2021-IT-01b) .....        | 77  |
| Kayles (2021-IT-02b) .....                | 80  |
| Golyók (2021-KR-02) .....                 | 83  |
| Hibakeresés (2021-LT-07) .....            | 86  |
| Hódia klánjai (2021-PH-03) .....          | 89  |
| Hídépítés (2021-PK-07) .....              | 92  |
| Koko (2021-SI-02) .....                   | 94  |
| Kulcstartók (2021-SK-01) .....            | 96  |
| Iskolai asztalok (2021-SV-01) .....       | 99  |
| Molnár Mona (2021-TR-06) .....            | 102 |
| A leghosszabb sorozat (2021-UA-01b) ..... | 104 |
| Dőőőő! a fa! (2021-UY-11) .....           | 107 |
| Mentsd meg a fát! (2021-UZ-02) .....      | 109 |
| Támogatóink, köszönetnyilvánítás .....    | 112 |



## FELADATOK

- Kishód:** 2021-HU-05c, 2021-IE-04, 2021-PK-07, 2021-CZ-06  
2021-DE-08a, 2021-SK-01, 2021-TR-06, 2021-HU-03  
2021-CY-03, 2021-DE-07, 2021-UA-01b, 2021-UY-11
- Benjamin:** 2021-DE-03, 2021-DE-08a, 2021-SK-01, 2021-TR-06, 2021-HU-03, 2017-AZ-02  
2021-CA-01b, 2021-CY-03, 2021-DE-07, 2021-KR-02, 2021-UA-01b, 2021-UY-11  
2021-AT-01, 2021-CH-04c1, 2021-HU-02, 2021-IN-05, 2021-LT-07, 2021-UZ-02
- Kadét:** 2021-CY-03, 2021-DE-03, 2021-DE-07, 2021-KR-02, 2021-UA-01b, 2021-UY-11  
2021-AT-01, 2021-CH-04c1, 2021-DE-08b, 2021-IN-05, 2021-LT-07, 2021-UZ-02  
2021-BE-02, 2021-CH-06, 2021-CZ-05, 2021-HU-02, 2021-ID-10, 2021-SV-01
- Junior:** 2021-AT-01, 2021-CH-04c1, 2021-DE-08b, 2021-IN-05, 2021-LT-07, 2021-UZ-02  
2021-BE-02, 2021-CH-06, 2021-CZ-05, 2021-HU-02, 2021-ID-10, 2021-SV-01  
2021-CA-02, 2021-CH-07a, 2021-CH-13, 2021-PH-03, 2021-SI-02, 2021-IT-01b
- Senior:** 2021-BE-02, 2021-CH-06, 2021-CZ-05, 2021-HU-02, 2021-ID-10, 2021-SV-01  
2021-CA-02, 2021-CH-07a, 2021-CH-13, 2021-PH-03, 2021-SI-02, 2021-IT-01b  
2021-AT-06, 2021-CZ-04, 2021-DE-05, 2021-HU-04, 2018-LT-04, 2021-IT-02b



## BINÁRIS KAPUK (2017-AZ-02)

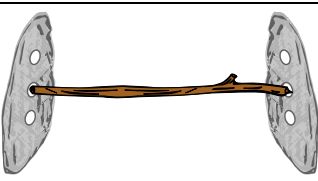
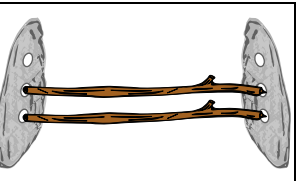
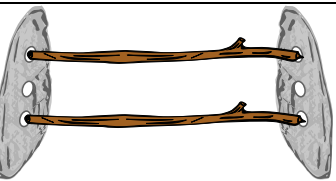
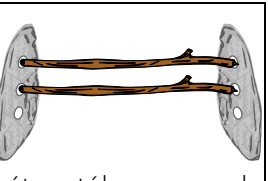
## BENJAMIN - KÖNNYŰ

A hódok gyakran meglátogatják egymást. De néha nincs otthon senki sem...

Ekkor hagynak egy üzenetet a kő kertkapujukon, mellyel a várható érkezésükről adnak hírt. A híradás az alábbiak szerint történik: Legfeljebb 3 faágat illesztenek be a kőkapu szemközti furataiba.

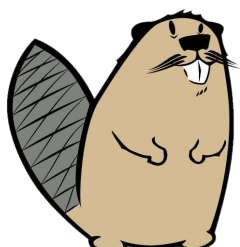


A hódok az alábbi jelentésekben állapodtak meg.

|                                                                                                                               |                                                                                                                            |                                                                                                                          |                                                                                                                                             |
|-------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
|  <p>Itthon vagyunk,<br/>gyere csak be.</p> |  <p>Már csak délben<br/>érünk haza.</p> |  <p>Sajnos csak este<br/>jövünk.</p> |  <p>Látogatóban vagyunk,<br/>éjfél körül érkezünk.</p> |
|-------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|

A hódok további üzenetekben is megállapodhatnak anélkül, hogy további faágat vagy furatot használnának.

### Hány további üzenettel találkozhatnak a látogató hódok?



## A helyes válasz a 4

Itt látható a lehetséges 8 üzenet, amelyet legfeljebb 3, a furatokba dugott faággal üzeni lehet. 4-nél ismert a kapu jelentése, a további 4-nél még nem.

### MIÉRT INFORMATIKA?

Ebben a feladatban a hódok a bináris számrendszert használják. Az egymással szemben található furatok az információhordozók, páronként 2 állapottal: vagy van benne faág, vagy nincs.

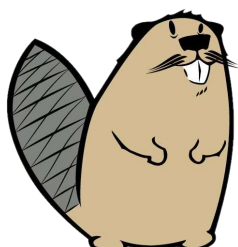
Matematikailag: A különböző üzenetek száma az alábbi módon adható meg: a furatok lehetséges állapotainak számát (2) emeljük a furatpárok darabszámára (3). tehát:  $2^3 = 2 \times 2 \times 2 = 8$ .

A hódok hagyományból tudják, hogy mit jelentenek az üzenetek. Ha mégis tévednének, nagy kár nem származna belőle. Az informatika világméretű hálózati rendszerében azonban mindenki mindenkivel közvetlen kommunikációs szomszéd és hibátlanul meg kell érteniük egymást.

A nagy szervezetek gondoskodnak a jelrendszerek szabványosításáról, standardizálásáról. A világ minden tájáról képviselt szakbizottságokban döntenek egy jel kinézetéről és annak jelentéséről. A legfontosabb számrendszereket az országgyűlések törvényileg szabályozzák, így érik el, hogy a világ összes számítógépe (is) jól megértse egymást.

### WEBOLDAL

[https://de.wikipedia.org/wiki/Normung#Internationale\\_Normung](https://de.wikipedia.org/wiki/Normung#Internationale_Normung)



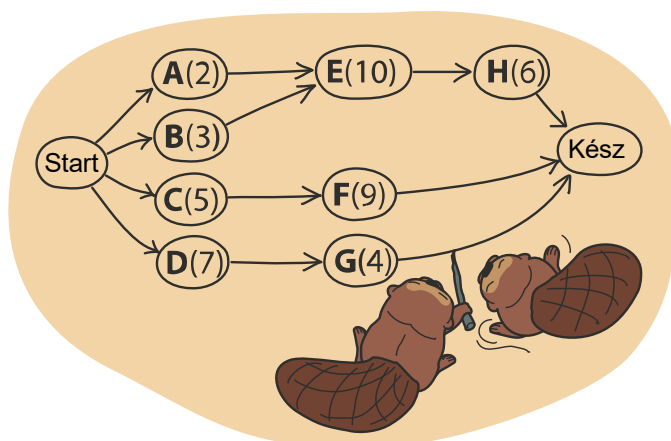


## KÉT HÓD DOLGOZIK (2018-LT-04)

## SENIOR – NEHÉZ

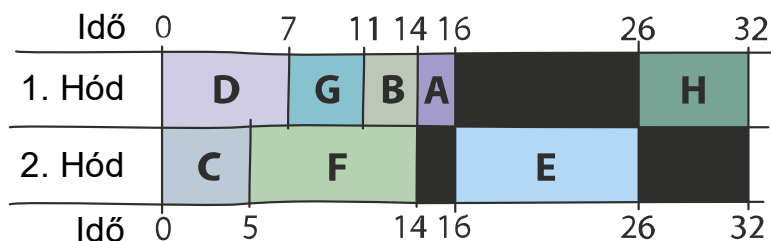
Két hód egy gátat épít és ehhez 8 feladatot kell elvégezniük: kivágni a fákat, legallyazni azokat, a törzseket a vízhez vinni stb...

Minden feladatot egy betűvel és mögötte zárójelben egy számmal jelöltünk. A zárójelben található számok azt mutatják, hány óra szükséges a munka elvégzéséhez. Egy feladat csak akkor kezdhető meg, ha bizonyos feladatok már befejezésre kerültek. A sorrendet az ábrán a nyilak mutatják.



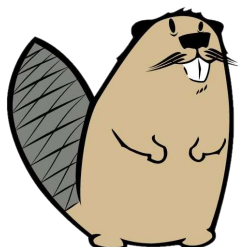
A hódok együtt dolgoznak, de mindegyik más-más feladaton.

A következő ábra a hódok beosztását mutatja.



Láthatod, hogy 32 óra alatt felépítik a gátat. De menne ez gyorsabban is!

**Hány óra alatt tudnának a hódok a gát megépítésével a leggyorsabban végezni? / Mennyi az a legrövidebb idő, ami alatt a hódok felépíthetik a gátat?**

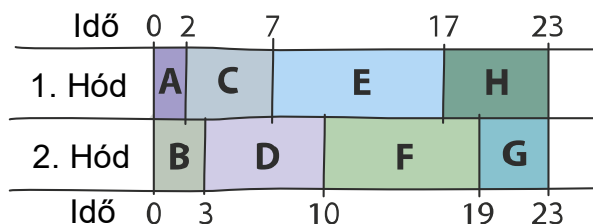


## A helyes válasz a 23

Legalább 23 órára szükségük van.

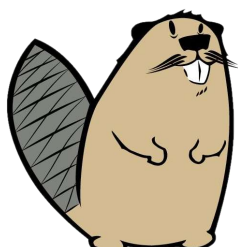
A feladatban szereplő kép a két hód beosztását mutatja. Láthatod, hogy az első hód egy hosszú (10 órás) szünetet tart. A második hód pedig összesen 8 órát nem dolgozik. Gyorsabban készen lehetnének, ha ezt az időt is munkával tölténék.

Ha odafigyelünk arra, hogy a két leghosszabb feladatot (F és E) ne egy hód végezze, máris egy jobb tervet tudunk összeállítani. Például egy megoldás arra, hogy 23 óra alatt hogyan végezhetnek:



23 óránál kevesebb idő alatt nem tudnak végezni, hiszen mindketten folyamatosan, szünet nélkül dolgoznak.

Ugyan össze tudnánk csoportosítani a feladatokat két 22 órás részre is  $(B(3)+E(10)+F(9))$  és  $A(2)+C(5)+D(7)+G(4)+H(6)$ , de ekkor az egymásra való várakozásnál keletkeznie kellene plusz időnek (hiszen a 2. hód még nem készül el az F feladathoz szükséges feladatokkal).



## MIÉRT INFORMATIKA?

A feladatban szereplő tervhez a hódok a következő szabályt alkalmazták: „Válassz a még el nem kezdett feladatok közül egy olyat, ami a legtöbb időt veszi igénybe”.

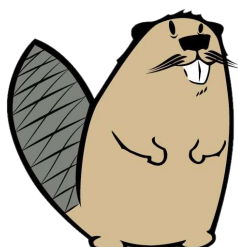
Az informatikában ezt a stratégiát mohó algoritmusnak (angolul greedy-nek) nevezik. Először azt a részfeladatot oldjuk meg, amelyik ránézésre a lehető legnagyobb lépést teszi meg a megoldás felé. Sok esetben ez egy jó stratégia, de időnként – mint pl. ebben a feladatban – nem működik olyan jól.

Tulajdonképpen csak egy biztosabb út létezik ahhoz, hogy megtaláljuk a feladat legjobb megoldását: minden lehetséges tervet kipróbálunk, melyek a megadott szabályoknak megfelelnek. Ezután eldöntjük, melyik a legmegfelelőbb. Ez nagyobb projektek esetében olyan sok lehetséges megoldást jelent, hogy túl sok idő lenne mindet megvizsgálni és a döntést ilyen módon meghozni. Ezért hasznosak pl. a mohó algoritmushoz hasonló stratégiák akkor is, ha nem mindig a legjobb megoldáshoz vezetnek.

Ennek a feladatnak a problémáját direkt állítottuk úgy össze, hogy ne működjön a megoldáshoz a mohó algoritmus. Ilyen haszontalan feladat megfogalmazása igazi művészet. Az elméleti informatikában célzottan keresnek ilyen haszontalan feladatmegfogalmazásokat (angolul „worst case”), hogy az algoritmusok időigényét jobban tudják tesztelni.

## WEBOLDAL

[https://hu.wikipedia.org/wiki/Moh%C3%B3\\_algoritmus](https://hu.wikipedia.org/wiki/Moh%C3%B3_algoritmus)



## MEGFIGYELÉS (2021-AT-01)

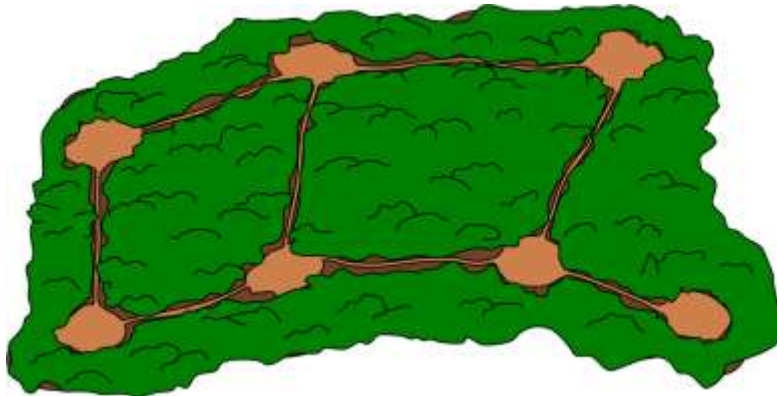
BENJAMIN - NEHÉZ

KADÉT – KÖZEPES

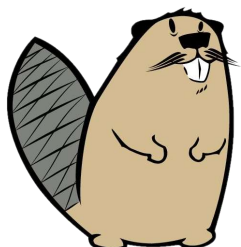
JUNIOR – KÖNNYŰ

A vadőrök szeretnék megfigyelni az erdei ösvényeken a vadak mozgását. Minden tisztásról az abból kiinduló és a következő tisztásig húzódó ösvényt látják.

**Hány tisztásra álljanak ki, ha a lehető legkevesebb számú tisztásról szeretnék megfigyelni az összes ösvényt?**

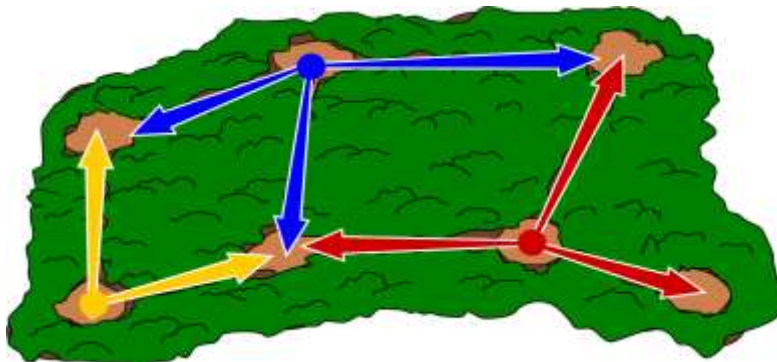


- A) 5 tisztásra
- B) 4 tisztásra
- C) 3 tisztásra
- D) 2 tisztásra



## A helyes válasz a C)

Az alábbi kép megmutatja azt a megoldást, amikor három tisztásról az összes ösvény megfigyelhető.



Összesen nyolc utat kell megfigyelni. Ha csak 2 tisztásról próbálnák megfigyelni az összes ösvényt, akkor kellene lennie olyan tisztásnak, ahonnan legalább négy ösvény indul ki. De ebben az erdőben nincs ilyen tisztás. Ezért 2 tisztás nem elég.

Legalább három tisztásról a képen mutatott módon megfigyelhető az összes ösvény. Valójában nincs más megoldás pontosan 3 tisztás mellett.

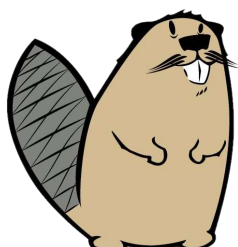
A nyolc megfigyelendő ösvény számából és abból, hogy nincsenek tisztítások, amelyekből háromnál több ösvény vezet ki, arra a következtetésre juthatunk: Minden tisztásról legalább két olyan utat kell megfigyelni, amelyeket a másik tisztásról nem figyelnek meg. Ezt a szabályt szem előtt tartva kezdhetjük el a tisztások kijelölését.

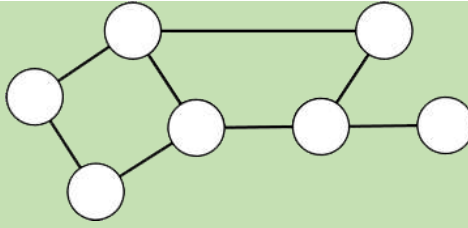
Annak érdekében, hogy a jobb alsó sarokban található, "zsákutca" tisztásra vezető ösvényt megfigyeljük, a vadőröknek a "zsákutca" bal végét jelentő tisztáson kell elhelyezkedniük, amint az az ábrán látható. Annak érdekében, hogy a jobb felső sarokban lévő vízszintes ösvényt lássuk, csak a középen felül lévő tisztás jön szóba. Végül marad a tisztás a bal alsó sarokban, melyből a még meg nem figyelt két ösvény látható. Tehát megvan az ábrán látható megoldásunk.

### MIÉRT INFORMATIKA?

A dolgok közötti kapcsolatokat (pl. a tisztítások közötti ösvényeket) úgynevezett gráfként ábrázolhatjuk.

A gráf csomópontokból (itt: a tisztások) és élekből (itt: ösvények), a csomópontok közötti vonalakkal áll. A feladat gráfja így néz ki:





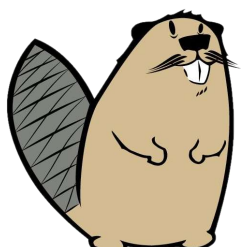
Ebben a hódfeladatban meg kell határoznunk a gráf csomópontjainak legkisebb számát, ahonnan a gráf minden éle elérhető.

Az informatikusok a csomópontok ilyen részhalmazát csomópontlefedésnek (minimális vagy optimális csúcslefedés) nevezik.

Ilyen csomópontlefedési problémákkal a mindennapi életben is találkozhatunk. Például amikor a legjobb helyeket keressük az utcai lámpák vagy a megfigyelőkamerák elhelyezéséhez.

WEBOLDAL

[https://hu.wikipedia.org/wiki/Maxim%C3%A1lis\\_f%C3%BCggetlen\\_cs%C3%BAcshalmaz](https://hu.wikipedia.org/wiki/Maxim%C3%A1lis_f%C3%BCggetlen_cs%C3%BAcshalmaz)



## TRUCHET (2021-AT-06)

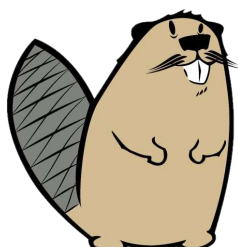
## SENIOR – NEHÉZ

A következő mintákat mindig egyféle (de időnként elforgatott) csempéből hoztuk létre. A felhasznált csempék nagytípusú jelennek meg.

**Melyik mintához hiányzik a csempe?**



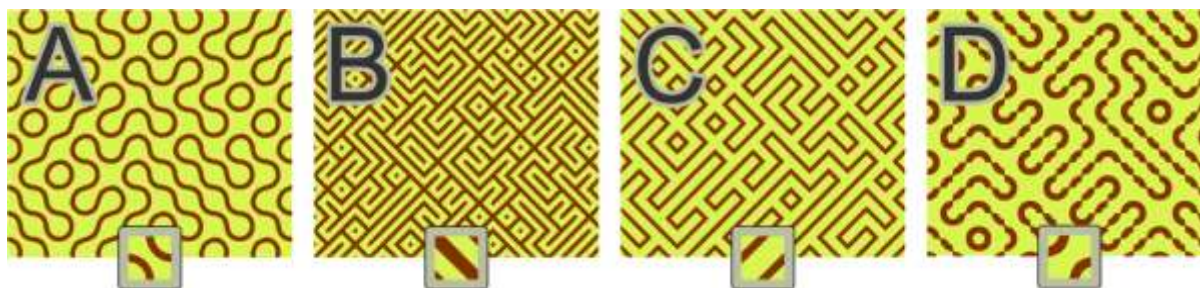
| A | B | C | D |
|---|---|---|---|
|   |   |   |   |





## A helyes válasz az A)

Az alábbi ábra mutatja, melyik mintát, milyen csempéből raktuk össze.



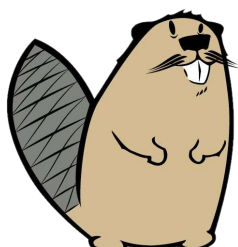
A  csempe az egyetlen olyan csempe, amelynek nincs 4 azonos oldala, azaz az egyes széleken nem ugyanott található barna, illetve sárga illesztés, kiindulási szín. Ennek eredményeként a kirakott minta nem olyan sima, mint a többi csempe esetében, ezért a D minta rendelhető hozzá.



A  csempe több és vastagabb barna résszel rendelkezik, mint a  csempe, ráadásul a sarkában található barna kis háromszög-csücskök összeillesztve egy négyzetet alkothatnak. Ezért a  csempe hozzárendelhető a B mintához.

A  csempe egyenes alakzataiból a C mintára lehet következtetni.

Így az A minta maradt meg, melyhez egy kerek formát  tartalmazó csempe szükséges.





## MIÉRT INFORMATIKA?

Ezeket a csempéket Sébastien Truchetről (\* 1657; † 1729) nevezték el, aki ilyen, és ehhez hasonló, nem szimmetrikus csempelapok elhelyezésének lehetőségeit vizsgálta.

A Truchet csempéknek nem feltétlenül kell 4 azonos oldallal rendelkezniük, mint a fenti 3 mintában.

Az a tény, hogy összetett mintákat lehet létrehozni nagyon egyszerű építőelemekkel, érdekes tulajdonság, amellyel újra és újra találkozunk az informatikában.

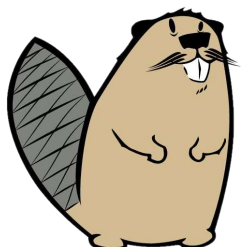
A Truchet csempéket a matematika és az informatika tudományterületein is tanulmányozzák, és számítógépes játékokban labirintusokat vagy dekorációkat készítenek a segítségével.

## WEBOLDALAK

[https://cifrapalota.blog.hu/2018/03/05/a\\_cementlapok\\_tortenetenek\\_rejtelye](https://cifrapalota.blog.hu/2018/03/05/a_cementlapok_tortenetenek_rejtelye)

[https://www.youtube.com/watch?v=2R7h76GoIJM&ab\\_channel=TheArtofCode](https://www.youtube.com/watch?v=2R7h76GoIJM&ab_channel=TheArtofCode)

<https://www.youtube.com/watch?v=tNYMBfAlhVs>



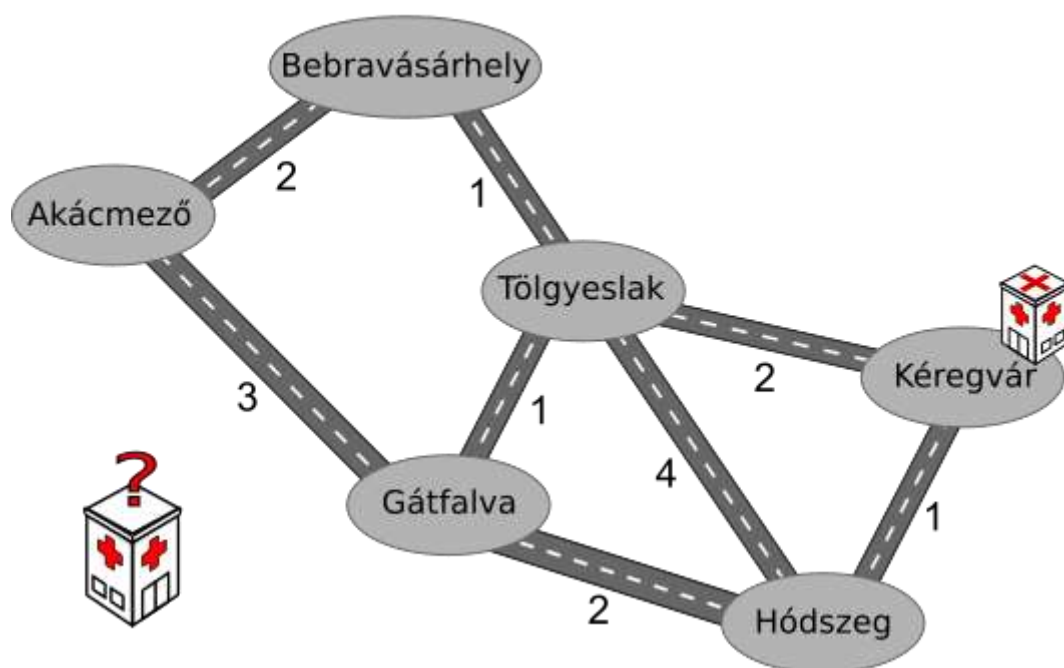
## KÖZPONTI KÓRHÁZAK (2021-BE-02)

KADÉT - NEHÉZ

JUNIOR – KÖZEPES

SENIOR – KÖNNYŰ

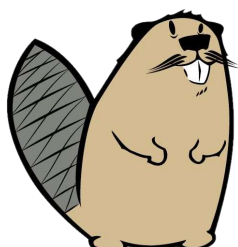
Hódföld régióban 6 város van. Ezek közül csak kettőben lehet kórház. Amikor a hódok úgy érzik, hogy betegek és segítségre van szükségük, a lehető leggyorsabban kórházba kell menniük. Az alábbi térkép Hódföldet és városait mutatja. A térképen lévő számok a két város közötti távolság megtételéhez szükséges időt jelzik órában megadva.



Egy kórház már van Kéregvár városában.

**Melyik városban építsenek a hódok egy második kórházat, ha azt szeretnék, hogy mindenhol a lehető leggyorsabban eljuthassanak a kórházak egyikébe?**

- A) Akácmező
- B) Bebravásárhely
- C) Tölgyeslak
- D) Gátfalva
- E) Hódszeg



## A helyes válasz a B)

Mivel egy kórház biztosan Kéregvárba kerül, ezért öt lehetőség van a második elhelyezésére. Minden lehetőségnél ellenőriznünk kell azt, hogy az egyes városokból mennyi idő szükséges a kórházakba való eljutáshoz. A következő táblázatban az oszlopok a két kórház lehetséges elhelyezését jelölik, a sorok pedig a legközelebbi kórház eléréséhez szükséges időt adják meg.

| Kiindulási város | A kórházak lehetséges elhelyezkedése |                            |                        |                      |                     |
|------------------|--------------------------------------|----------------------------|------------------------|----------------------|---------------------|
|                  | Akácmező és Kéregvár                 | Bebravásárhely és Kéregvár | Tölgyeslak és Kéregvár | Gátfalva és Kéregvár | Hódszeg és Kéregvár |
| Akácmező         | 0                                    | 2                          | 3                      | 3                    | 5                   |
| Bebravásárhely   | 2                                    | 0                          | 1                      | 2                    | 3                   |
| Tölgyeslak       | 2                                    | 1                          | 0                      | 1                    | 2                   |
| Gátfalva         | 3                                    | 2                          | 1                      | 0                    | 2                   |
| Hódszeg          | 1                                    | 1                          | 1                      | 1                    | 0                   |
| Kéregvár         | 0                                    | 0                          | 0                      | 0                    | 0                   |
| <b>Maximum</b>   | <b>3</b>                             | <b>2</b>                   | <b>3</b>               | <b>3</b>             | <b>5</b>            |

Amint az az utolsó sorban látható, a Bebravásárhelyen történő elhelyezés a legjobb megoldás, hiszen a legmesszebb lakó hódok számára is csak legfeljebb két órát vesz igénybe, hogy eljussanak bármelyik városból egy tetszőleges kórházba.

## MIÉRT INFORMATIKA?

Ez a feladat a gráfcentrumok megkeresésének a problémája, melynek megoldására több algoritmust is kidolgoztak már.

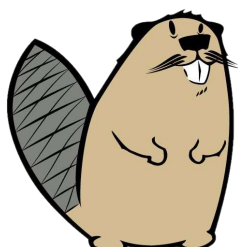
A megoldáshoz a régió és úthálózata ábrázolható gráfként úgy, hogy a városok megfelelnek a gráf csúcsainak, az őket összekötő utak az éleknek, míg az utazáshoz szükséges idők pedig az egyes élek súlyainak. Jelen esetben a probléma abból áll, hogy  $k$  csomópontot kell választunk a gráfból (számszerint kettőt) úgy, hogy minimális legyen a gráf bármely csomópontjától a  $k$  kiválasztott csomópontokhoz való eljutás szükséges ideje.

Ez a probléma nagyon gyakori, amikor megpróbálják kiválasztani, hogy hol helyezzenek el fontos létesítményeket, mint a tűzoltóság, az iskola, a rendőrség stb. (vagy a kórház, ahogy a példában is szerepel). A minimalizálandó kritérium lehet a létesítmények eléréséhez szükséges idő, az utazási távolság vagy bármi más, ami elengedhetetlen az adott létesítmény hatékony működéséhez.

## WEBOLDALAK

Vertex  $k$ -center probléma: [https://en.wikipedia.org/wiki/Vertex\\_k-center\\_problem](https://en.wikipedia.org/wiki/Vertex_k-center_problem)

Gráf centruma: [https://hu.wikipedia.org/wiki/Gr%C3%A1f\\_centruma](https://hu.wikipedia.org/wiki/Gr%C3%A1f_centruma)



## PÖTYIK (2021-CA-01B)

## BENJAMIN - KÖZEPES

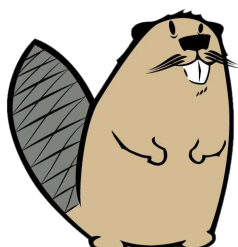
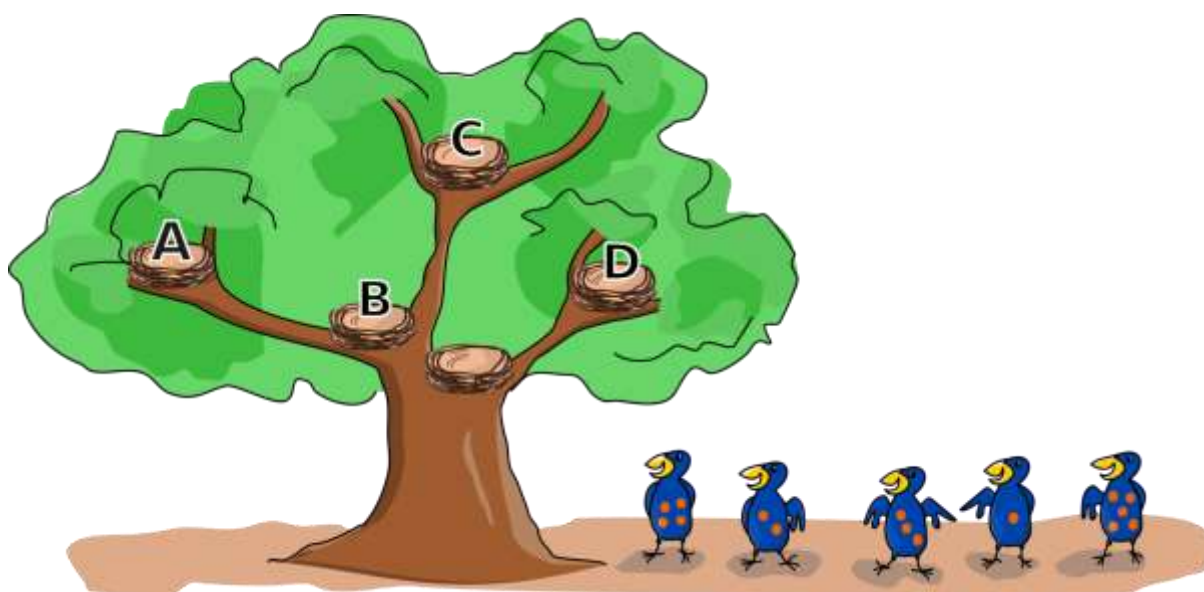
A pötyik olyan madarak, amelyek hasán pontok találhatók.

Egy fa mellett öt pötyi várakozik. Egymás után – sorrendben balról jobbra – beköltöznek a fán látható üres fészkekbe. A négypontos hasú pötyi az első.

Minden pötyi ugyanolyan szabályokat követve költözik be. A fa tövéből indul és a következő lépéseket hajtja végre amíg nem talál egy üres fészket:

1. Addig mászik felfele (arccal a fa felé), amíg fészket nem talál.
2. Ha a fészek üres, beköltözik és ott marad.
3. Ha a fészek már foglalt, tovább mászik, mégpedig:
  - a. ha a fészekben lévő pötyi hasán több pont van, mint neki, akkor balra mászik.
  - b. ha ugyanannyi vagy kevesebb pontja van, akkor pedig jobbra.

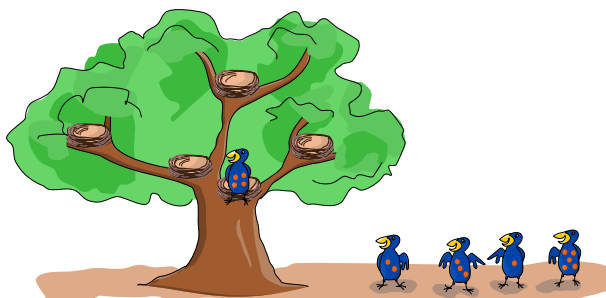
**Melyik fészkekbe kerül a sor végén álló pötyi öt ponttal a hasán?**



## A helyes válasz a D jelű fészek

A pötyik a következőképpen foglalják el a fészkeket:

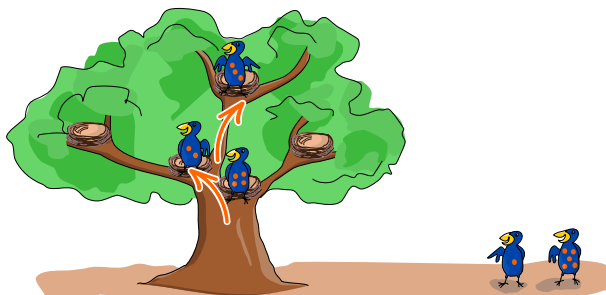
A pötyi a 4 ponttal a hasán beköltözik a legalsó fészekbe



A második pötyi hasán két pont van, így a legalsó fészekben ülő pötyi négy pontja miatt (hiszen a  $4 > 2$ ), balra indul tovább. Beköltözik a következő üres (B jelű) fészekbe.

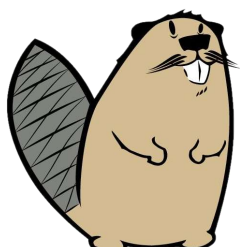


A harmadik pötyi hasán három pont van. Ezért az első fészeknél, ahol a négy pontos pötyi fészkel, balra mászik tovább. A következő fészekben, a kétpontos pötyinél pedig jobbra, hiszen a  $3 > 2$ .

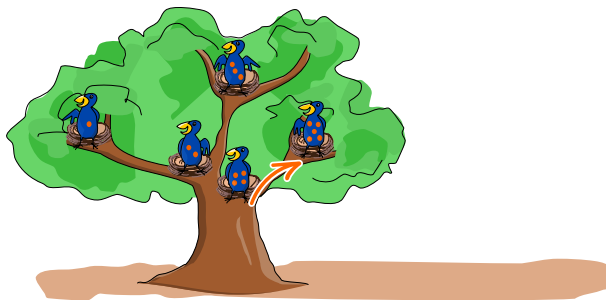


A következő fészek (C jelű, a legmagasabban elhelyezkedő) üres, így oda beköltözik.

A negyedik pötyi hasán egy pont található. Így az első fészeknél (négy pontos pötyi) balra, majd a következő fészeknél (kétpontos pötyi) ismét balra mászik tovább. Ezután talál egy üres fészket (A jelű), amibe beköltözik.



Az utolsó pötyi hasán öt pont található, így az első fészeknél jobbra mászik tovább (hiszen  $4 < 5$ ). Itt el is éri a D jelű fészket, melybe beköltözhet.



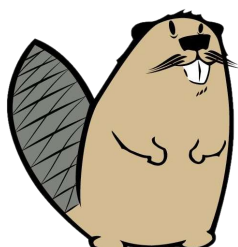
### MIÉRT INFORMATIKA?

Ha a pötyik ezzel az eljárással költöznek be a fészkekbe, annak van egy érdekes előnye: egy adott pötyi gyorsan megtalálható. Ha a keresett pötyi hasán kevesebb pont található, mint az adott fészkekben ülő hasán, akkor a fa bal részén kell tovább nézni. Ellenkező esetben jobbra kell vizsgálni. Minden alkalommal, amikor egy madarat megvizsgálunk, a felére csökken az a terület, amelyet még vizsgálnunk kell. Ezért gyorsan megtalálhatjuk a pötyit.

Az adatok rendszerezésének ezt a módját bináris keresési fának nevezzük. A "bináris" szó a latin "bi" szóból származik, melynek jelentése "kétszer". Mert egy ág végén (ahol fészkek vannak a feladatban), legfeljebb két kisebb ág folytatódik. A bináris keresési fákat számítógépes programokban használják, amikor sok adatot kell gyorsan megtalálni. Általában sokkal nagyobbak, mint a feladatban szereplő kis fa. Különbség is van: a feladatban lévő fába öt helyre öt számot (pötyit) kellett elhelyeznünk. A valóságban bármikor további adatokat adhatunk hozzá egy bináris keresési fához. Minden egyes ág behelyezésekor egy új ág kerül az ág végére, ezáltal megnövelve a fát. Mivel a bináris keresési fa változhat, dinamikus adatstruktúrának nevezzük.

### WEBOLDALAK

<https://www.geeksforgeeks.org/binary-search-tree-data-structure/>  
[https://hu.wikipedia.org/wiki/Fa\\_\(adatszerkezet\)](https://hu.wikipedia.org/wiki/Fa_(adatszerkezet))



## PÓKHÁLÓ (2021-CA-02)

JUNIOR – NEHÉZ

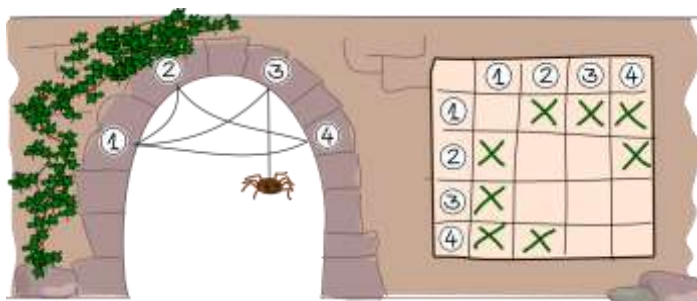
SENIOR – KÖZEPES

Tekla pókhálót vizsgál. Kitalált egy módszert, amellyel leírhatja a hálók pontos szerkezetét.

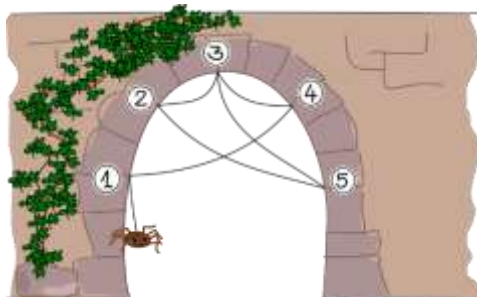
A háló végpontjait 1-től N-ig számozza és egy négyzetrács mezőit használja a következő séma szerint:

- Ha van egy szál, amely összeköti az 1. végpontot a 2. végponttal, akkor az 1. oszlop és a 2. sor mezőjét "x" jelöli.
- Az 1. végpontot a 2. végponttal összekötő szál a 2. végpontot az 1. végponttal is összeköti, tehát a 2. oszlop és az 1. sor mezőjébe is kerül "x".

Itt látható egy pókháló, és Tekla leírása:

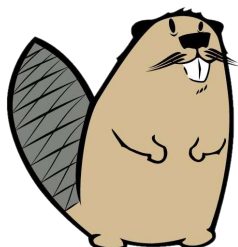


Tekla talált egy újabb pókhálót.



**Melyik négyzetrácsal írja le?**

| A | B | C | D |
|---|---|---|---|
|   |   |   |   |





## A helyes válasz az A)

Ha követjük a leírt szabályokat, akkor az A) választ kapjuk.

A B válaszban egy kapcsolat helytelenül lett megrajzolva (az 1. végpont és a 2. végpont között mindkét irányban), és egy nem szerepel (2. végpont és 5. végpont mindkét irányban).

Az itt használt módszernél valójában nem lehet "x" az átlóban. Ez a végpont és saját maga között lenne egy kapcsolat (szál), ami bizonyos hálókbán megengedett, de a pókhálókbán nem fordul elő. A C válaszban azonban 2 ilyen kapcsolat is van (az 1. végpontban és a 4. végpontban).

A hálók minden ábrázolásának szimmetrikusnak kell lennie a bal felső saroktól a jobb alsó sarokig (úgy, hogy a szimmetriatengely az átló). A D válasz tartalmazza a kapcsolatot a 2. végponttól az 5. végpontig, de hiányzik az ennek megfelelő másik kapcsolat az 5. végponttól a 2. végpontig.

### MIÉRT INFORMATIKA?

A pókháló tekinthető egy gráfnak, amely az informatikában széles körben használatos.

A gráf csomópontokból (a hálózat végpontjai) és élekből (a végpontok közötti szálakból) áll. A gráfokat többek között az objektumok és az objektumok közötti kapcsolatok ábrázolására is használják. Például egy gráf ábrázolhatja a baráti kapcsolatokat a közösségi médiában vagy repülőjáratokat az országok között.

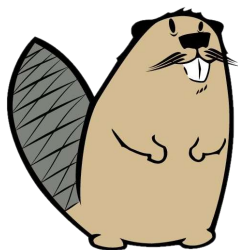
Ez a feladat bemutatja, hogyan lehet ábrázolni egy gráfot egy négyzetrácsban. Bizonyos tulajdonságok, például a gráf pontos kinézete, elvesznek a folyamat során. Sok esetben azonban a gráf geometriai tulajdonságai nem is érdekesek, hanem csak a szerkezete. A gráf bizonyos tulajdonságai így is megmaradnak: pl. létezik-e egy bizonyos él? Hány él kapcsolódik egy adott csomóponthoz?

A bemutatott lehetőség csak egy a sok közül egy hálózat szerkezetének leírására. A módszer nem túl gazdaságos, mert minden kapcsolat mindkét irányban mentésre kerül, ami nem lenne szükséges, és a szabad átlós mezők is feleslegesek. Ennek a módszernek azonban így az az előnye, hogy a megjelenítési hibák részben felismerhetők.

A bemutatott ábrázolási formát szomszédsági mátrixnak nevezzük.

### WEBOLDAL

[https://hu.wikipedia.org/wiki/Szomsz%C3%A9ds%C3%A1gi\\_m%C3%A1trix](https://hu.wikipedia.org/wiki/Szomsz%C3%A9ds%C3%A1gi_m%C3%A1trix)





## EPERTOLVAJ (2021-CH-04C1)

BENJAMIN - NEHÉZ

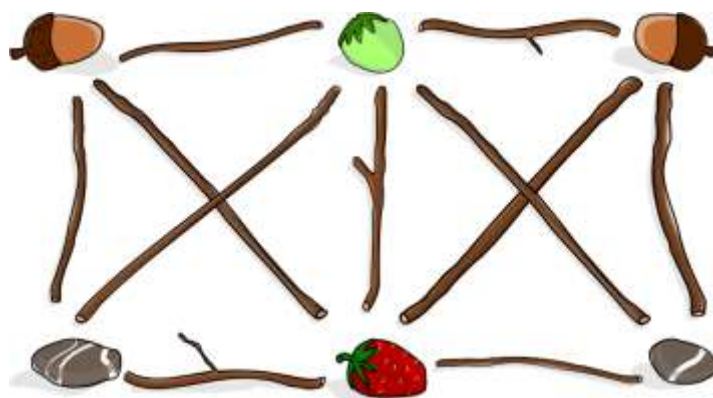
KADÉT - KÖZEPES

JUNIOR – KÖNNYŰ

Anna makkokból, mogyorókból, kavicsokból és eperből egy műalkotást rakott ki a gyepre.

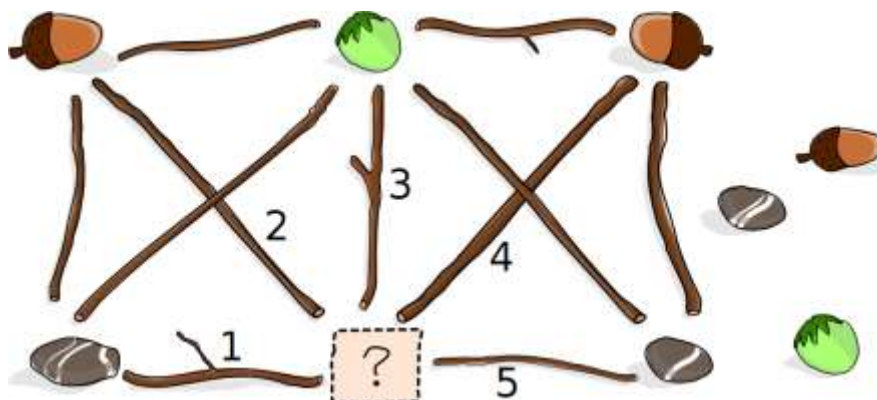
Ezután ágakat tett az egyes dolgok közé a következő szabályt követve: Egy ág nem fekszik közvetlenül két azonos dolog között - például nem tett ágat két makk közé.

Ez a kész mű:

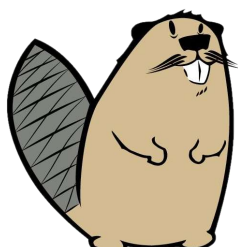


Anna bátyja megette az epret. Nem is maradt belőle. Hogy elrejtse tettét, egy másik dolgot tett az eper helyére, és eltávolított pontosan egy ágot, hogy Anna szabálya ne sérüljön.

**Mit tett az eper (?) helyére és melyik ágot távolította el?**



- A) Mogorót és a 3-as ágot
- B) Mogorót és az 5-ös ágot
- C) Makkot és a 2-es ágot
- D) Kavicsot és az 5-ös ágot

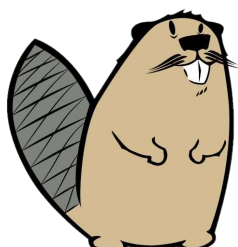


Ha az eper helyére mogyorót tett, a középen lévő 3-assal jelölt ág megsérti Anna szabályát: Két azonos dolog között van, nevezetesen két mogyoró között. Ezért ezt az ágat el kell távolítani. Többet azonban nem.

- Ha az eper helyét makk váltja fel, a 2. és 4. ágat kell eltávolítani.
- Ha az epret kaviccsal helyettesíti, az 1. és 5. ágat kell eltávolítani.

A gráf minimális színekkel történő színezésének, a klikkek elemzésének problémája sokféleképpen használható. Például a sportversenyek megtervezésénél, az ülésterv elkészítésénél és még a Sudoku-rejtvény megoldásában is.

[https://hu.wikipedia.org/wiki/Gr%C3%A1fok\\_szerkezet%C3%A9re](https://hu.wikipedia.org/wiki/Gr%C3%A1fok_szerkezet%C3%A9re)  
[https://hu.wikipedia.org/wiki/Klika\\_\(gr%C3%A1felm%C3%A9let\)](https://hu.wikipedia.org/wiki/Klika_(gr%C3%A1felm%C3%A9let))



## TÖMÖR MEGJELENÍTÉS (2021-CH-06)

KADÉT - NEHÉZ

JUNIOR – KÖZEPES

SENIOR – KÖNNYŰ

Hód Hedvig 1-esekkel és 0-akkal szeretne betűket kódolni. Mivel úgy találta, hogy az F és az E betűk gyakrabban fordulnak elő, ezért ezeknek rövidebb kódot határozott meg. Így az F, E, É, L, S és G betűkhöz az alábbi kódtáblázatot készítette:

| Betű | F | E  | É    | L    | S    | G    |
|------|---|----|------|------|------|------|
| Kód  | 1 | 00 | 0010 | 0110 | 1010 | 1110 |

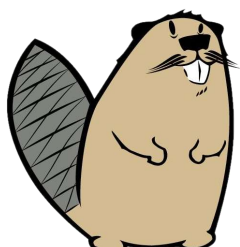
Ezután a következő üzenetet küldte el Hugónak:

1 0 0 1 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 1 1 1 0 0 0

Hugó tudja, hogy az üzenet utolsó betűje E.

**Mi lehetett Hedvig üzenete?**

- A) FÉLESÉGE
- B) FELESÉGE
- C) ELESÉGE
- D) FÉLÉSE



## A helyes válasz az A)

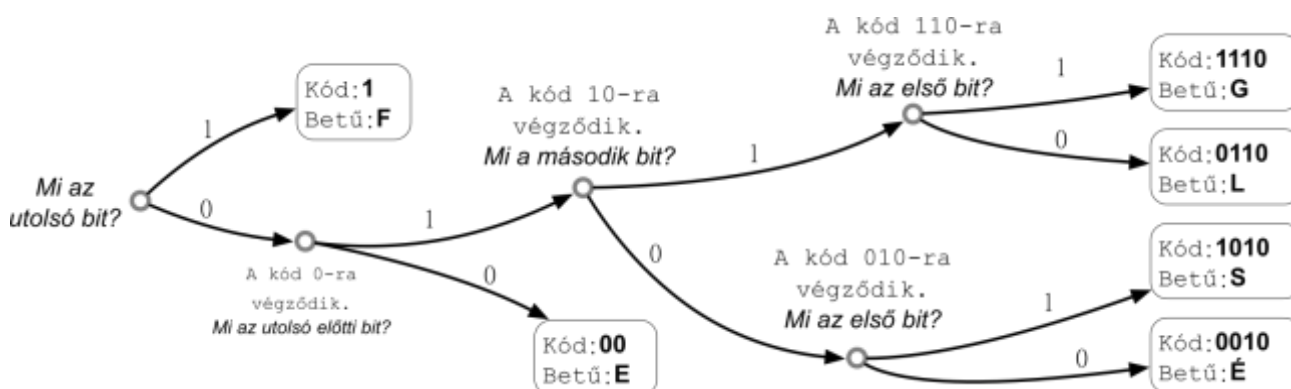
Az üzenet rekonstruálásához meg kell találni a módot arra, hogy az egész üzenetet betűire tudjuk bontani. Ha balról (hátról) indulunk, nem olyan könnyű ezt megtenni: általában kétértelműségekbe ütközünk.

Próbáljuk meg előlről: meglehetősen könnyen azonosíthatjuk, hogy az első betű F és 1 -nek felel meg. (Ugyan az S és a G is 1-gyel kezdődik, de egyik sem 00-val folytatódik, így a kezdés egyértelmű). Ezért a C válaszlehetőséget ki is ejthetjük.

A második betű már gondot okozhat: lehet E, „00”, vagy É, „0010”. Még nem lehet biztosan tudni. Az üzenet további vizsgálatánál azonban ez is kiderül: ha rosszul választunk a másodiknál, később elakadunk, és rájövünk, hogy az egyetlen lehetőség az É volt.

(Az E esetében a következő betű az 1, az 10 vagy az 1001 lenne, de ezek közül csak az 1, az F létező. Innen a következő betű akkor ismét E kellene, hogy legyen. Hasonlóan tovább gondolkodva eljutunk a FEFEFFE-ig ami után a 0, 01 0101 következne. De egyikhez sincs betű hozzárendelve, így visszalépve látjuk, hogy a második betűnek az E nem felelt meg. Tehát a B válaszlehetőség sem megfelelő.)

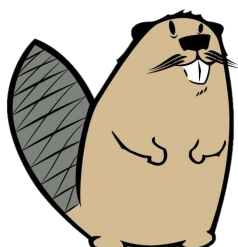
Esetünkben azonban rájöhettünk, hogy ha az üzenet végéről kezdjük a megfejtést, akkor sem kell soha „találgatni” egy-egy betű dekódolásához. Ez azért van, mert a kód suffix-free (utótag-mentes): azaz nincs kódszó, amely ugyanarra végződik, mint egy másik.



### MIÉRT INFORMATIKA?

Minden olyan objektumot, amellyel a számítógép dolgozik, bitek (0-sok és 1-esek) sorozataként kell leírni. Ez igaz a szövegekre is.

Az ember mindig azt várja, hogy az eredeti objektum rekonstruálható a bináris ábrázolásából, de ez csak akkor lehetséges, ha soha nem fordul elő, hogy két vagy több különböző objektumot ugyanaz a bináris érték reprezentál. Az informatikusok egyik feladata, hogy kidolgozzanak olyan kódrendszereket, amelyek hatékonyan megvalósíthatók és egyértelműen rekonstruálhatóak belőlük az eredeti objektumok (például szövegek).



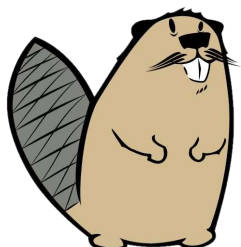
Ha valaki tömöríteni akar egy szöveget (a lehető legrövidebb bináris ábrázoláshoz), akkor jó stratégia, ha rövidebb bináris kódokat veszünk a gyakoribb betűkhöz és hosszabb kódokat használunk a ritkán előforduló betűk esetében. Ekkor azonban ügyelni kell arra, hogy olyan kódokat válasszunk, amelyek garantálják az egyértelmű dekódolást (az eredeti szöveg rekonstrukcióját). Ilyen módszer a Huffman-kódolás is.

Korábbi hód feladatokban már találkozhattál prefix kódokkal, amikor a egy betű kódjának eleje sohasem egyezhet meg más betű kódjával.

Itt a suffix kódokkal dolgoztunk, amikor a kódsorok végét egyeztetjük a betűk esetében: azaz egy betű kódja sem végződhet olyan kóddal, ami már létező, betűhöz rendelt kód.

WEBOLDAL

<https://hu.wikipedia.org/wiki/Huffman-k%C3%B3d%C3%A1s>.



## SAPKA, KORONG (2021-CH-07A)

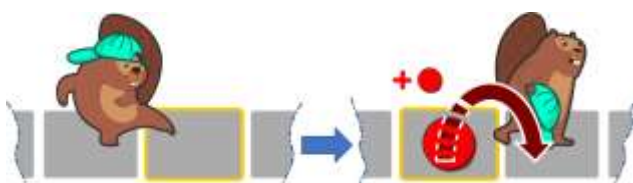
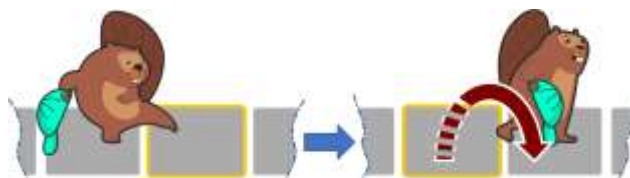
JUNIOR – NEHÉZ

SENIOR – KÖZEPES

A hódok egyik kedvenc játéka a "Sapka, korong".

Ebben a hód mindig balról jobbra lépdel (→) négyzetről négyzetre és másképp viselkedik attól függően, hogy a sapka a kezében vagy a fején van.

Ha egy hód üres négyzetre lép és a sapka a kezében van, akkor csak átlép az üres négyzetre.



Ha a hód üres négyzetre lép, de a sapka a fején van, akkor letesz egy korongot arra a négyzetre, amin állt, átlép az üres négyzetre és a sapkát a kezébe veszi.

Ha a hód egy olyan négyzetre lép, amin van egy korong és a sapka a kezében van, akkor felveszi a korongot, átlép a következő mezőre és a fejére teszi a sapkát.

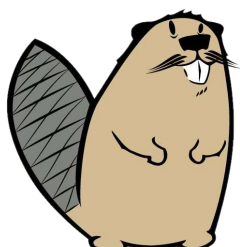


Ha a hód egy olyan négyzetre lép, amin van egy korong és a sapka a fején van, akkor csak átlép az üres négyzetre (nem veszi fel a korongot és nem veszi kézbe a sapkát sem).

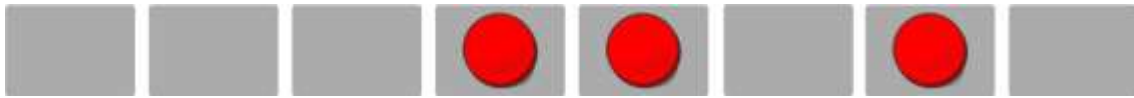
Kezdetben a hód sapkával a kezében áll a négyzetek bal oldalán:



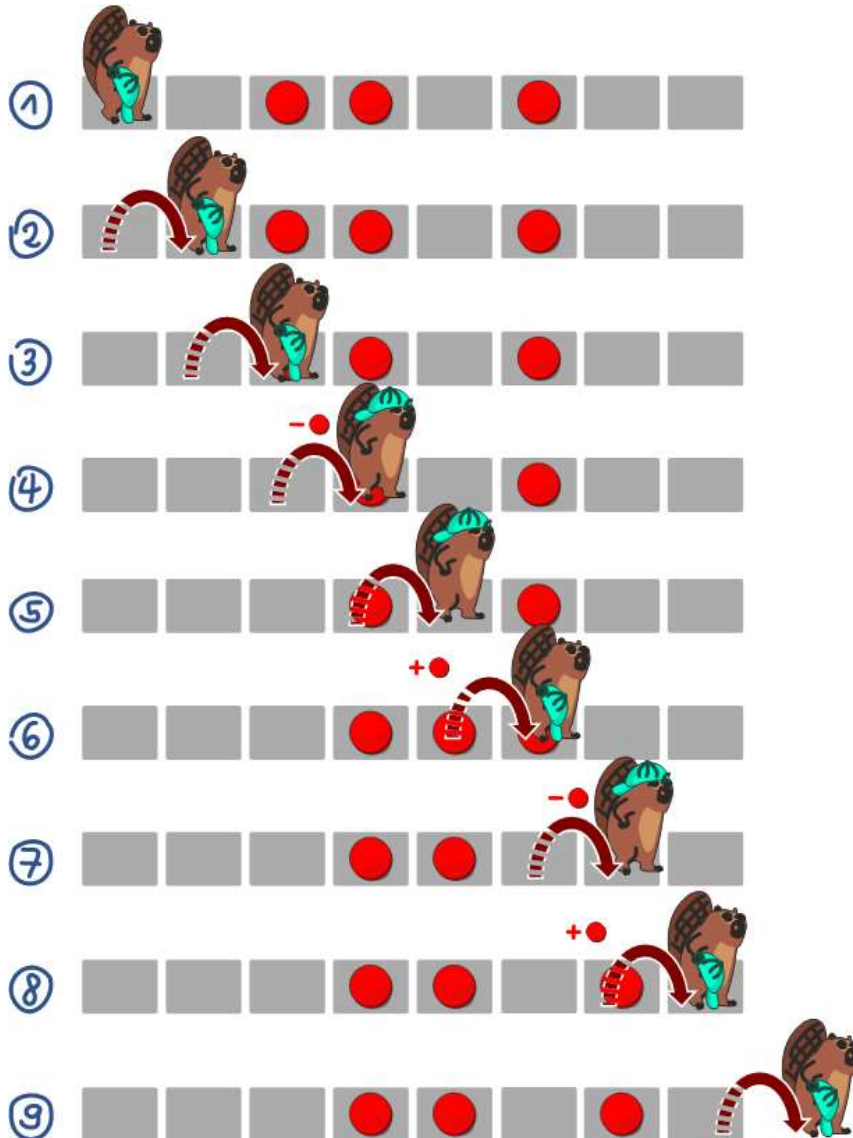
**Hol találhatók a korongok amikor a hód lelépett az utolsó négyzetről is?**



A helyes válasz

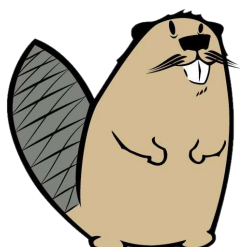


A megoldás lépésről lépésre:



#### MIÉRT INFORMATIKA?

Az informatikában a viszonylag egyszerű műveletek gyakran érdekes eredményekhez vezetnek. Ez a feladat jó példa erre. A fenti algoritmus lényege, hogy a hódnak két különböző állapota van (a fején van-e a sapka vagy sem), és üres vagy foglalt négyzeteket talál útközben. A szabályok alapján ez egy művelet, amely egy bináris biteltolódásra emlékeztet, azaz amikor egy bináris számot



megszorzunk 2-vel. Bármilyen egyszerű is ez a példa, a Turing-gép néhány alapvető elemét tartalmazza.

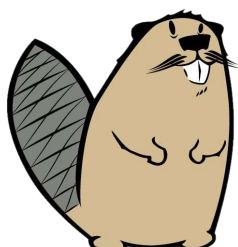
A Turing-gép a következőkből áll:

- **Szalag**, amelyről le lehet olvasni és rá lehet írni. Példánkban ezek azok a négyzetek, amelyeken a hód mozog.
- **A szimbólumok véges ábécéje**, néha szándékosan 0-ra és 1-re korlátozva. Példánkban a korong 1-et, az üres négyzet pedig 0-t szimbolizál.
- **Olvasó**, illetve író fej, amely képes leolvasni a szalagról szimbólumokat és ráírni azokat. Valamint mindkét irányban mozoghat a szalagon. Példánkban a hódnak olvasó / író fej szerepe van, de csak egy irányba mozog.
- **Állapotok véges halmaza**. Esetünkben csak azok az állapotok vannak, amelyeknél a "hód fején van a sapka" vagy a "hód kezében van a sapka".
- **Szabályok**: mi történik az állapottól függően, amikor egy bizonyos szimbólumot olvasunk le a szalagról? A lehetséges műveletek a következők: állapotátmenet, szimbólum írása a szalagra és az olvasó / író fej balra vagy jobbra mozgatása.

#### WEBOLDALAK

<https://hu.wikipedia.org/wiki/Turing-g%C3%A9p>

<https://hu.wikipedia.org/wiki/Bitm%C5%B1velet#Biteltol%C3%A1sok>





## GYÜMÖLCS TÍZÓRAIRA (2021-CH-13)

JUNIOR – NEHÉZ

SENIOR – KÖZEPES

Egy négyfős családban apa, anya, Dóra és Robi este mindig négy tízórais dobozt készít össze, mindegyikbe más-más gyümölcsöt tesznek: almát, banánt, narancsot vagy görögdiányt. A dobozokat ezután egyszerűen egymásra állítják a hűtőszekrényben.

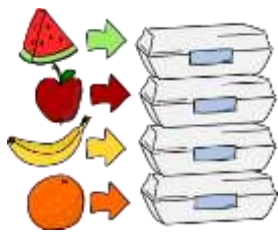
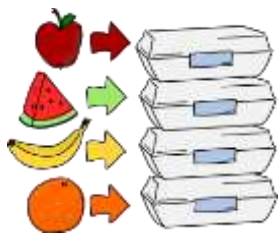
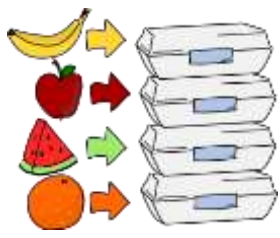
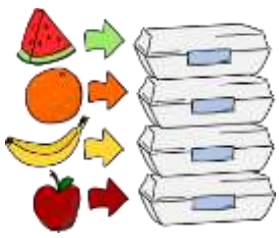
Reggel a családtagok még mindig nagyon álmosak, és amikor elhagyják a lakást, egyszerűen csak kiveszik a legfelső dobozt a hűtőből, anélkül, hogy belenézniene.

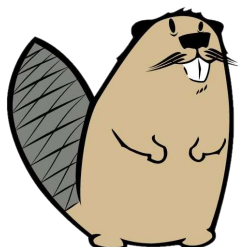
Nem tudni pontosan, hogy milyen sorrendben hagyják el a lakást, de az biztos, hogy anya mindig Dóra előtt és apa mindig utolsóként.

A családtagok többféle gyümölcsöt is szeretnek. A táblázat azt mutatja, hogy ki melyiket:

|      | <br>alma | <br>banán | <br>narancs | <br>dinnye |
|------|-------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| apa  |                                                                                           |                                                                                            | ✓                                                                                             |                                                                                               |
| anya | ✓                                                                                         |                                                                                            | ✓                                                                                             | ✓                                                                                             |
| Dóra | ✓                                                                                         | ✓                                                                                          | ✓                                                                                             |                                                                                               |
| Robi | ✓                                                                                         | ✓                                                                                          |                                                                                               | ✓                                                                                             |

**Milyen sorrendben állítsák egymásra a dobozokat a hűtőszekrénybe, hogy mindenki biztosan olyan gyümölcsöt vigyen magával, amit szeret?**

| A                                                                                   | B                                                                                   | C                                                                                    | D                                                                                     |
|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
|  |  |  |  |



## A helyes válasz az A)

Csak egy módja van a dobozok egymásra állításának, hogy mindenki olyan gyümölcsöt vegyen ki a hűtőszekrényből, amit szeret:

Apa csak a narancsot szereti és mindig utolsónak távozik. Tehát a narancs az alsó dobozba kerül.

Robi első, második vagy harmadik. Anya mindig Dóra előtt megy, így ha Robi távozását ismerjük, ehhez képest megadhatjuk, mikor mennek el a lakásból.

A táblázat mutatja a lehetséges eseteket:

|   |      |      |      |
|---|------|------|------|
| 1 | anya | anya | Robi |
| 2 | Dóra | Robi | anya |
| 3 | Robi | Dóra | Dóra |
| 4 | apa  | apa  | apa  |

Anya, Dóra és Robi is mehet másodikként. Ezért felülről a második dobozba olyan gyümölcsöt kell tenni, amelyet mindhárman szeretnek. Ez pedig az alma.

Tehát csak a banán és a dinnye kerülhet a legfelső dobozba. Mivel anya nem szereti a banánt, és ő néha elsőként is távozik, ide kell beletenniük a dinnyét. A banán a harmadik dobozba kerül, ami megfelelő, mert Dóri és Robi is szereti.



## MIÉRT INFORMATIKA?

Az informatika számos területén fontos a helyes sorrend meghatározása: Sok számításhoz köztes eredményekre van szükség, amelyeket először meg kell határozni, mielőtt az végső számítás eredményét megkapnánk. Ha a számítási lépéseket különböző számítógépeken hajtják végre, gondos tervezés nélkül úgynevezett „holtpontok” (deadlocks) merülhetnek fel. Ezek olyan helyzetek, amikor két vagy több folyamat vár egymásra, és így a program soha nem ér véget.

A rossz sorrend rendszerint hibákhoz is vezet. Például, ha a  $Z = (A+B) * (A-B)$  képletet kell kiszámítani, akkor ezt a következő lépésekre kell felbontani:

Határozzuk meg A-t

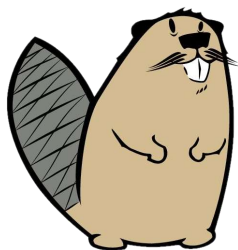
Határozzuk meg B-t

Számítsuk ki  $X = A + B$

Számítsuk ki  $Y = A - B$

Számítsuk ki  $Z = X * Y$

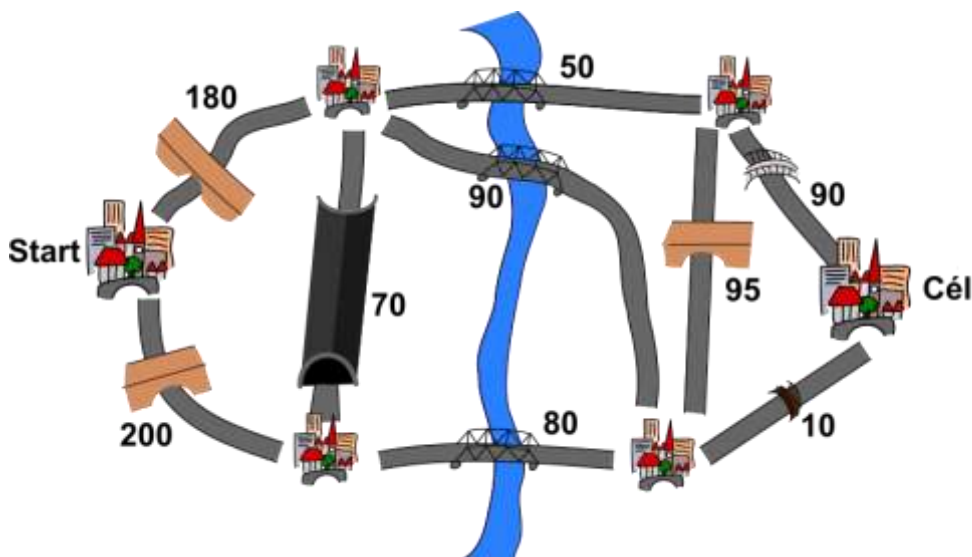
Ha az első számítást anélkül hajtánánk végre, hogy a B-t meghatároztuk volna, akkor a programunk hibás eredményt adhat (pl. azzal az értékkel számol, ami a lefoglalt memóriaterületen található, vagy egy állandó kezdőértéket használ).



## MAGASSÁGKORLÁTOZÁS (2021-CY-03)

KISHÓD - NEHÉZ  
BENJAMIN - KÖZEPES  
KADÉT - KÖNNYŰ

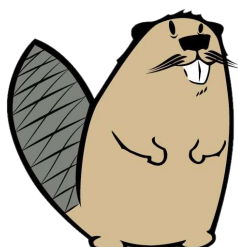
Hódvár városai között az áruk szállítását az alábbi útvonalakon oldják meg.



A hidak és az alagutak miatt bizonyos szakaszokon korlátozni kell a teherautók magasságát. Az autóutak melletti számok jelölik azt, hogy az adott helyen mekkora lehet a teherautók maximális magassága tölgyméterben mérve.

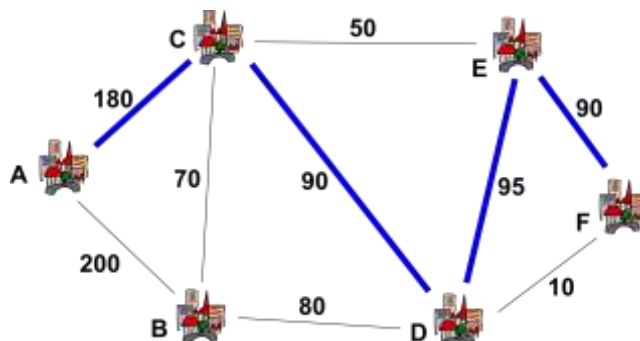
**Mekkora lehet annak a teherautónak a maximális magassága, amely a Start és a Cél felirattal jelölt város között közlekedik?**

- A) A: 90 tölgyméter
- B) B: 80 tölgyméter
- C) C: 70 tölgyméter
- D) D: 50 tölgyméter



## A helyes válasz az A)

Ahhoz, hogy a Starttól a Célig eljussanak, a teherautónak át kell haladnia valahol a folyón. Ehhez csak három utat használhat, ahol a határérték 50, 90 és 80 tölgyméter. Ez azt jelenti, hogy a legjobb esetben is csak maximum egy 90 tölgyméter magasságú teherautó küldhető el az elejétől a végéig.



Lehetséges, hogy egy 90 tölgyméter magasságú teherautót küldjenek az elejétől a végéig? Igen! Ha egy teherautó a fenti útvonalon közlekedik, akkor a magassági határértékek 180, 90, 95 és 90 tölgyméter lesznek. A legkisebb közülük a 90 tölgyméter. Ez azt jelenti, hogy egy 90 tölgyméter magasságú teherautót küldjenek végéig a két város közti úton.

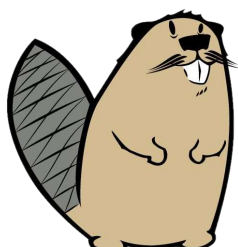
A probléma megoldásának másik (jóval hosszabb) módja az, hogy gondosan ellenőrizzük az összes lehetőséget, akár a Starttól kezdve a Célig, akár a Céltól visszafelé.

## MIÉRT INFORMATIKA?

Ezt a problémát a *számítógépes hálózaton* (pl. wifi vagy kábelek egy épületben) küldött *adatok* korlátozásai inspirálták. A *routerek* és az *útválasztó algoritmusok* irányítják az adatforgalmat: az *adatcsomagok* szegmensről szegmensre ugrálnak, ahol "átirányítják" azokat a következő szegmensre. Az egyik fontos szempont a *sávszélesség*, ami korlátozza, hogy az adatcsomagok milyen gyorsan továbbíthatók a szegmens mentén egy adott időszakban. Ez hasonlít arra, a korlátozásra, amelyet a hidak és az alagutak magassága támaszt a teherautók felé.

## WEBOLDAL

<https://hu.wikipedia.org/wiki/%C3%9Atv%C3%A1laszt%C3%B3>



## ÖNVEZETŐ PADLÓROBOT (2021-CZ-04)

## SENIOR – NEHÉZ

Hód Herold épített egy pályát a padlórobotjának. A robot a képen látható helyről indul és a vonalak mentén képes mozogni. Három jelzés van (◊, ○ és ✕) a vonalakon, amelyek eldöntik, hogy milyen irányban kell haladnia a következő kereszteződésben. A robot nem érheti el a  jelzést.

A különböző jelzések különböző utasításokat jelentenek az alábbiak közül:

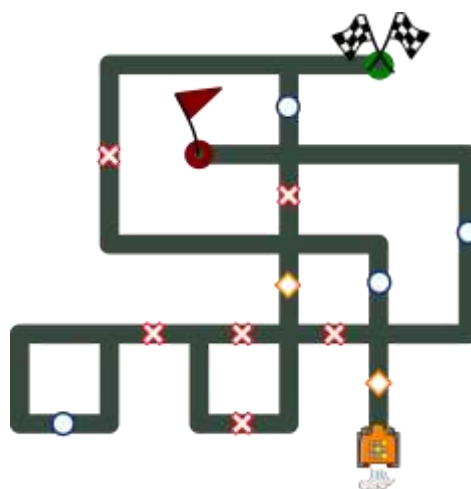
**fordulj balra** a következő kereszteződésnél;

**fordulj jobbra** a következő kereszteződésnél;

**menj egyenesen** a következő kereszteződésnél.

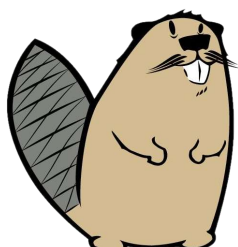
Sajnos nem tudjuk, melyik jelzés mit jelent.

A képen látható nyíl azt mutatja, hogy a robot hogyan fordulna, két különböző irányból érkező, ha a háromszög jelzés azt jelenti, hogy "fordulj balra".



**Milyen utasításokat rendeljünk hozzá az egyes jelzésekhez, hogy a robot elérje a  jelzést?**

- A) ✕ = fordulj jobbra, ○ = menj egyenesen, ◊ = fordulj balra
- B) ✕ = fordulj balra, ○ = menj egyenesen, ◊ = fordulj jobbra
- C) ✕ = fordulj jobbra, ○ = fordulj balra, ◊ = menj egyenesen
- D) ✕ = menj egyenesen, ○ = fordulj jobbra, ◊ = fordulj balra
- E) ✕ = fordulj balra, ○ = fordulj jobbra, ◊ = menj egyenesen
- F) ✕ = menj egyenesen, ○ = fordulj balra, ◊ = fordulj jobbra





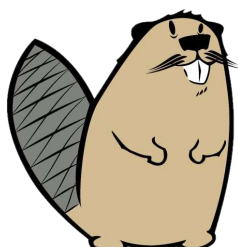
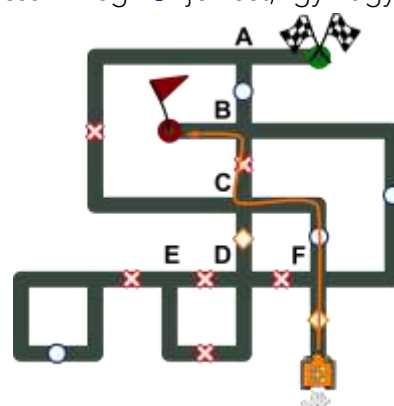
|                                                           |                                                           |                                                           |
|-----------------------------------------------------------|-----------------------------------------------------------|-----------------------------------------------------------|
|                                                           |                                                           |                                                           |
| = menj egyenesen,<br>= fordulj jobbra,<br>= fordulj balra | = fordulj balra,<br>= fordulj jobbra,<br>= menj egyenesen | = menj egyenesen,<br>= fordulj balra,<br>= fordulj jobbra |

Egy másik stratégia az, hogy nyomon követjük azt az utat, amelyet a robot megtehet, és az utasításokat hozzárendeljük a jelzésekhez. Ha a robot eléri a piros zászlót vagy belép egy hurokba (újra meg újra ugyanazt az utat teszi meg), illetve az utasítás nem alkalmazható következetesen a jelzésekre, visszalépünk az előző lépésig (ebben az esetben a kereszteződésig) és megpróbálunk egy másik útvonalat.

Amikor a robot eléri az F kereszteződést, három lehetősége van: forduljon balra, menjen egyenesen vagy forduljon jobbra.

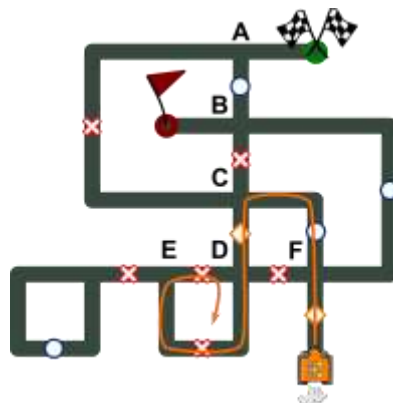
[PRÓBA 1] Tegyük fel, hogy egyenesen megy. Ami azt jelenti, hogy "menj egyenesen" utasításnak felel meg. A robot továbbhalad a C-kereszteződés felé. Mivel a "menj egyenesen" utasítás már hozzá lett rendelve a jelhez, és a robot most látta csak meg jelzést, így vagy balra fordul, vagy jobbra fordul.

- [PRÓBA 1.1] Tegyük fel, hogy a robot jobbra fordul a C kereszteződésben, ami azt jelenti, hogy a jelzéshez a "fordulj jobbra" utasítás tartozik, míg az jelzéshez pedig a "fordulj balra". A robot most balra kanyarodna a B kereszteződésben, elérve a jelzést. Ez a megoldás számunkra nem jó, ezért visszalépünk az előző C kereszteződéshez.



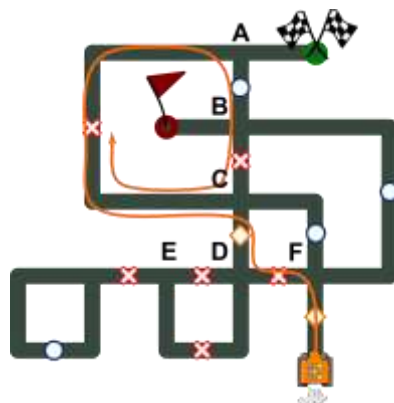


- [PRÓBA 1.2] Tegyük fel, hogy a robot balra fordul a C kereszteződésben ami azt jelenti, hogy a  jelzéshez a "fordulj balra" utasítás tartozik, míg az  jelzéshez pedig a "fordulj jobbra". A D kereszteződésben a robot egyenesen haladt, a  jelzésnek megfelelően, majd eléri az E kereszteződést. Az E-nél jobbra fordulna, amikor meglátja az  jelzést. A D kereszteződésben ismét jobbra fordulna, így egy hurokba lépne. Számunkra ez nem jó megoldás, ezért visszalépünk a C kereszteződéshez, hogy ellenőrizzük, lehetséges-e más út. Mivel nincs más értelmezési módja a jelzéseinknek, arra a következtetésre jutottunk, hogy még a korábbi feltételezésünk is helytelen volt, és visszalépünk az F kereszteződésig, és megpróbálunk egy alternatív útvonalat.

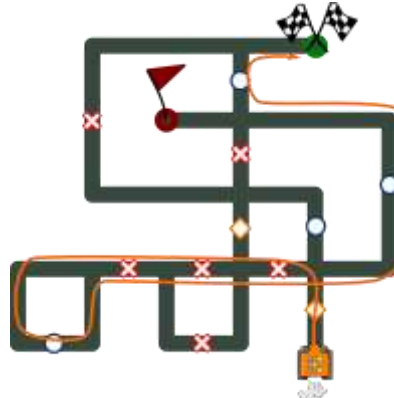


[PRÓBA 2] Tegyük fel, hogy a robot balra fordul az F kereszteződésben, ami azt jelenti, hogy a  jelzéshez a "fordulj balra" utasítás tartozik.

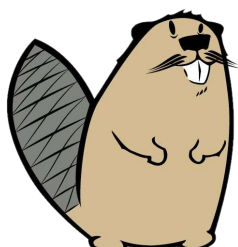
- [PRÓBA 2.1] Amikor a robot eléri a D kereszteződést, ismét 3 lehetőség áll rendelkezésére. Balra nem fordulhat, hiszen az azt jelentené, hogy az  és a  jelzéshez is ugyanaz az utasítás tartozik. A robot jobbra fordulhat a D-nél, ami azt jelenti, hogy az  jelzés "fordulj jobbra", és ezért a  jelzés azt jelenti, hogy "menj egyenesen". D-ből a robot továbbmegy be. A C-nél balra fordul, ahogy látja a  jelzést. Az A kereszteződésben jobbra fordulna, miután látta az  jelzést. A B kereszteződésben egyenesen a C kereszteződésig folytatódna az útja a  jelzés miatt. A C kereszteződésben ismét jobbra fordulna és egy hurokba kerülne az  jelzés miatt. Számunkra ez nem jó megoldás, ezért visszalépünk egészen a D kereszteződésig (ahol úgy döntöttünk, hogy jobbra fordulunk) és keresünk egy másik utat.
- [PRÓBA 2.2] Most tegyük fel, hogy a robot egyenesen halad (ne feledjük, hogy nem fordulhat balra), tehát  jelzés a "menj előre", és így a  jelzés a "fordulj jobbra" utasítást fogja jelenteni. Így az utat követve látható, hogy a robot eléri a  jelzést. Ezért a jelzések helyes értelmezése:  "fordulj balra",  "menj egyenesen", és  "fordulj jobbra".



C-



az





## MIÉRT INFORMATIKA?

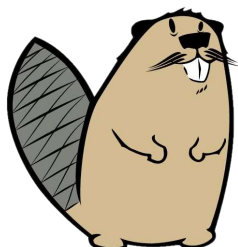
A magyarázatban javasolt első stratégiát a "nyers erő" módszerének (*brute force*) nevezik. Ez azt jelenti, hogy minden lehetőséget mérlegelni és ellenőrizni kell, amíg meg nem találjuk a kívánt eredményt. Néha sok lehetőség adódhat, és mindegyik figyelembevétele nagyon időigényes lehet, ezért más stratégiákat kell alkalmazni. A második stratégiát visszalépéses keresésnek (*backtracking*) nevezik. Ebben a stratégiában az ember fokozatosan építi fel a lépéseket, és elhagyja azokat, amiről megállapítható, hogy nem vezet helyes megoldáshoz. A teljes kipróbálással szemben a visszalépéses keresés előnye, hogy nem kell a lépéseket az elejétől fogva újra végrehajtani, mivel csak arra a lépésre tér vissza, ahol a rossz lépés előtt az utolsó döntést hozta, és onnan folytatja.

Az adott problémára a brute force jobban működhet, mivel kevés változónk (jelzések) van és kevés út áll rendelkezésre a robotnak. Nagyobb és komplexebb problémák megoldására azonban, ahol számottevőbb a változók száma és jóval több megoldási lehetőségünk van, a visszalépéses keresés jobb, hatékonyabb és elegánsabb megoldást nyújt.

Az olyan rejtvények, mint a Sudoku, könnyedén megoldhatók a visszalépéses kereséssel.

## WEBOLDAL

[https://hu.wikipedia.org/wiki/Brute\\_force-t%C3%A1mad%C3%A1s](https://hu.wikipedia.org/wiki/Brute_force-t%C3%A1mad%C3%A1s)



## ELLENŐRZŐ BIZOTTSÁG (2021-CZ-05)

KADÉT - NEHÉZ

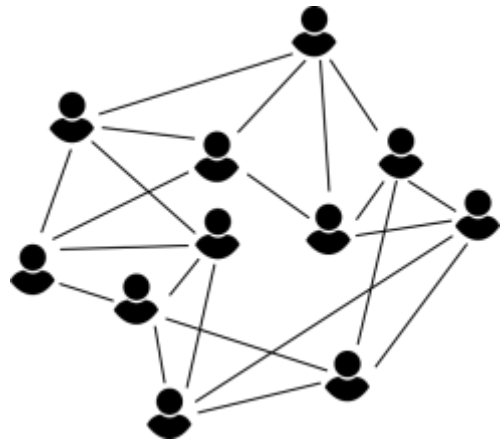
JUNIOR – KÖZEPES

SENIOR – KÖNNYŰ

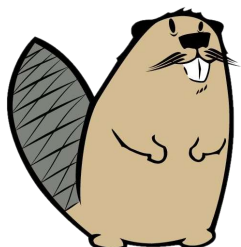
Hódia város tanácstagjai különböző (munkahelyi, rokoni, párt vagy üzleti) kapcsolatban állhatnak egymással.

A városi tanácsnak 11 tagja van. Bármely két tag, akiknek valamilyen kapcsolata van egymással, egy vonallal van összekötve az ábra szerint.

A városi tanács Ellenőrző Bizottsága a tanács olyan tagjaiból állhat, akik nem állnak egymással semmilyen kapcsolatban.

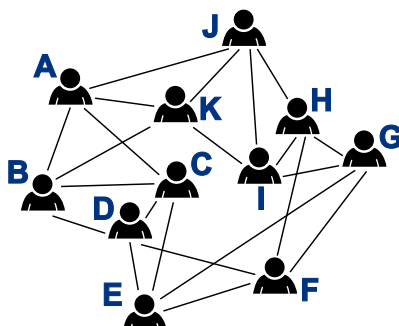


**Hány tagja lehet legfeljebb (maximum) az Ellenőrző Bizottságnak?**

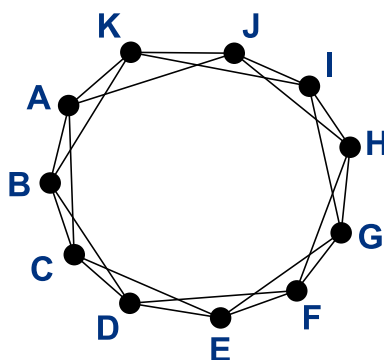


## A helyes válasz a három

Nevezzük el az egyes tanács tagokat betűkkel:



Rendezzük át a grafikont úgy, hogy még jobban láthatóak legyenek a kapcsolatok: azaz, egyértelműen kitűnjön, hogy mindenkinek 4 kapcsolata van.



Például a C pont csak A, B, D, E kapcsolatokkal rendelkezik.

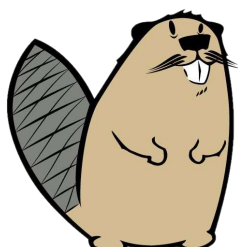
Bárhol elkezdhetjük az Ellenőrző Bizottság tagjainak kiválasztását, mert a gráfnak (a rajzolt ábránknak) nincsenek különleges pontjai.

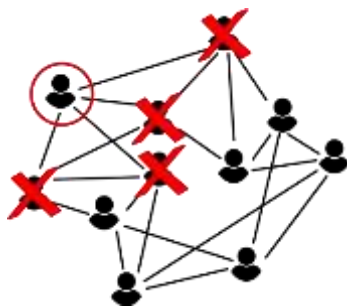
Ha az A tagot választjuk, akkor a következő legközelebbi tag, akinek nincs kapcsolata A-val az óramutató járásával ellentétes irányban, az D. A következő, D-vel való kapcsolat nélküli G, majd a következő ilyen J. De J-nek van kapcsolata A-val, így ő már kiesik és csak 3 (A, D, G) tag független egymástól.

Ugyanannyi Ellenőrző Bizottsági tagot kapunk, amikor egy másik tetszőleges pontból indulunk el, függetlenül attól, hogy milyen irányba megyünk.

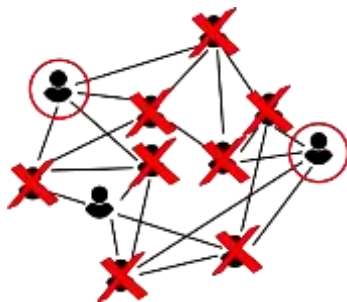
Ha nem jut eszünkbe az ábra átrendezése, egyszerű kihúzással is nekikezdhethetünk a feladatnak. Mivel mindenki négy kapcsolattal rendelkezik, nincs különleges, kiemelt tagunk, így bárkivel kezdhethetünk.

Kijelölünk egy tagot (piros karika) és kihúzzuk (piros X) azokat, akikkel kapcsolata van:

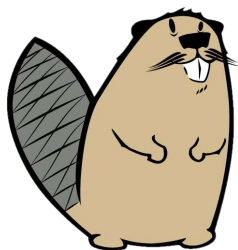
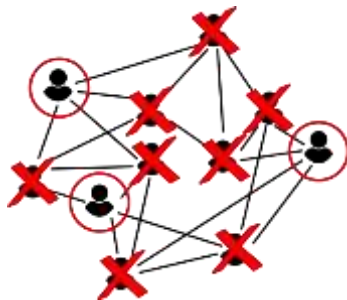




Ezután kiválasztunk valakit a megmaradt tagok közül és neki is kihúzzuk a kapcsolatait:



Már látszik, hogy egy tagunk maradt, akinek a kapcsolatait már mind kizártuk (a másik két kiválasztott tag miatt), így ő lesz a harmadik kiválasztott tagunk.



## MIÉRT INFORMATIKA?

A városi tanács tagjai közötti kapcsolatokat gráf segítségével modellezzük. Ez csomópontokból áll, élekkel összekötve (általában pontokat összekötő vonalakként ábrázolva). A feladat gráfja és a magyarázat gráfja is ugyanazt ábrázolja.

A gráf absztrakt szerkezet, és akkor hasznos, ha az összefüggésekre akarunk koncentrálni: hangsúlyozza a fontos jellemzőket (ki kivel áll kapcsolatban), de kihagyja a nem fontos részleteket (mik az érintett személyek/tárgyak, milyen összefüggés alapján kapcsolódnak egymáshoz).

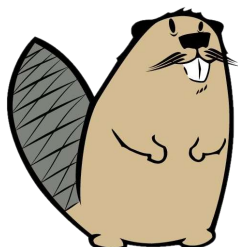
A számítógépek nagyon hatékonyan tudnak dolgozni a gráfokkal, ezért az informatikusok sokszor használják és rengeteg algoritmust gondoltak már ki arra, hogy az egyes válaszokat hogyan lehet megtalálni (pl. legtöbb kapcsolat, legtöbb egymástól független csomópont megtalálása, ...).

A gráf terminológiáját használva ebben a hód feladatban a gráf olyan független halmazait, azaz csomópontok részhalmazait szeretnénk megtalálni, amelyek közül bármely kettőt nem köt össze egy él. Azaz a gráf függetlenségi számát keressük, amely a független csúcsok maximális száma.

## WEBOLDALAK

[https://hu.wikipedia.org/wiki/F%C3%BCggetlen\\_cs%C3%BAcshalmaz](https://hu.wikipedia.org/wiki/F%C3%BCggetlen_cs%C3%BAcshalmaz)

[https://hu.wikipedia.org/wiki/Maxim%C3%A1lis\\_f%C3%BCggetlen\\_cs%C3%BAcshalmaz](https://hu.wikipedia.org/wiki/Maxim%C3%A1lis_f%C3%BCggetlen_cs%C3%BAcshalmaz)



## ÉRMEPIRAMIS (2021-CZ-06)

KISHÓD - KÖNNYŰ

Gabinak 6 érméje van:

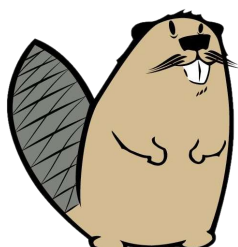


Ezeket piramis alakban egymásra fektette:



Vajon melyik érmét fektette az asztalra negyedikként Gabi?

| A                                                                                   | B                                                                                   | C                                                                                    | D                                                                                     |
|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
|  |  |  |  |



## A helyes válasz a B)

Az érmeket Gabi a következő sorrendben fektette egymásra:



Mindegyik érmét fedí legalább egy másik érme, így a megoldás megtalálható például az utolsóként legfelül elhelyezett érmétől indulva. Az utolsó érme, amelyet semelyik másik érme sem

fed, az a  érme. Az az érme amelyiket csak ez az egy legfelső fed az: . És a negyedik lerakott érme, melyet csak ez a két előző érme fed, a .

## MIÉRT INFORMATIKA?

A képen látható érmék sorrendben vannak elhelyezve. Ugyanezt a hatást láthatod, ha képet rajzolsz a számítógépen: például rajzolsz egy kört, majd két pontot, majd egy ívelt vonalat, akkor smiley-t kapsz. Ha utoljára rajzoltad volna meg a kört, akkor a két pont és ívelt vonal a kör mögé került volna.



A számítógépek általában egymást követő utasítások végrehajtásával működnek. A legtöbb számítógépes program úgy van megírva, hogy először egy, majd egy másik művelet történik. Tehát egy smiley rajzolásához használt számítógépes program így nézhet ki:

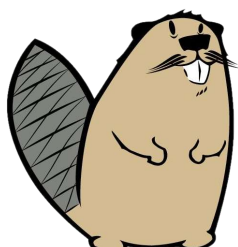
```
rajzolj kört (5,5) pontba 5 sugárral  
rajzolj pontot (2,7) pontba  
rajzolj pontot (7,7) pontba  
rajzolj balra ívelt vonalat (2,2) ponttól (7,2) pontig
```

Természetesen az egymás utáni utasítások egymás utáni végrehajtása nem az egyetlen dolog, amit egy program vagy egy számítógép tud.

A bonyolultabb programokhoz szükség van döntések meghozatalára bizonyos feltételek alapján, az ismétlésekre, és segít a gyakran használt programrészek különálló, úgynevezett alprogramokba helyezése is.

## WEBOLDALAK

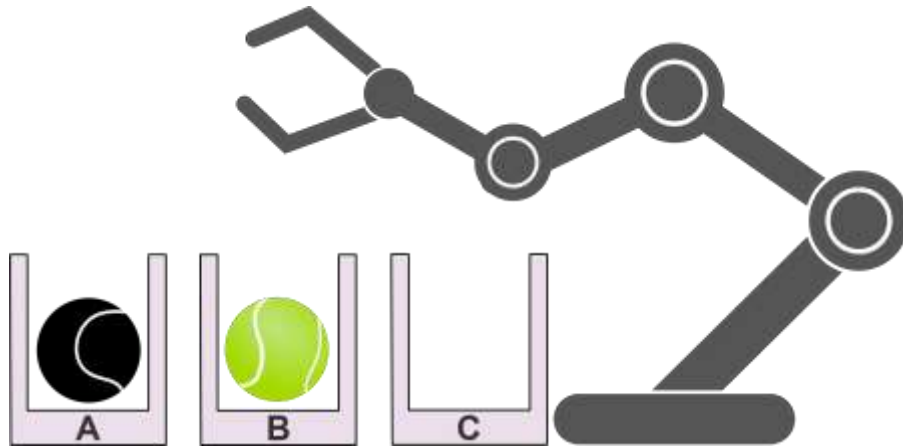
[https://wiki.prog.hu/wiki/Vez%C3%A9rl%C3%A9si\\_szerkezet](https://wiki.prog.hu/wiki/Vez%C3%A9rl%C3%A9si_szerkezet)  
[https://hu.wikipedia.org/wiki/Struktur%C3%A1lt\\_programoz%C3%A1s](https://hu.wikipedia.org/wiki/Struktur%C3%A1lt_programoz%C3%A1s)



## ROBOTKAR (2021-DE-03)

BENJAMIN - KÖNNYŰ

KADÉT - KÖNNYŰ



Van két labdánk: egyik az "A" rekeszben, másik a "B" rekeszben. A harmadik, "C" rekesz üres.

A robotkar az alábbi utasításokat hajtja végre egymás után:

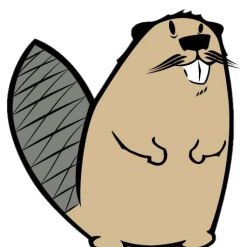
Vedd fel a labdát az "A"-ból és tedd bele a "C"-be.

Vedd fel a labdát a "B"-ből és tedd bele az "A"-ba.

Vedd fel a labdát a "C"-ből és tedd bele a "B"-be.

**Az utasítások végrehajtása után, az alábbi állítások közül melyik lesz igaz?  
Több választ is megjelölhetsz!**

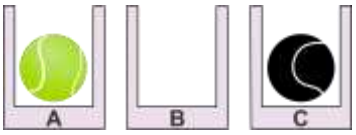
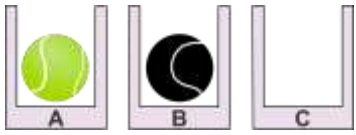
- A) A labdák helyet cseréltek.
- B) Két labda van az "A" rekeszben.
- C) Két labda van a "B" rekeszben.
- D) Az "A" rekesz üres.
- E) Semmi sem változott, mindkét labda visszakérült az eredeti helyére.





## A helyes válasz az A) és az E)

A) és E) a helyes válaszok. Lássuk az állapotokat minden egyes utasítás után:

| 1. utasítás után                                                                  | 2. utasítás után                                                                  | 3. utasítás után (végső)                                                            |
|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
|  |  |  |

## MIÉRT INFORMATIKA?

Ez a feladat a számítógép által vezérelt algoritmusok végrehajtását szemlélteti. A feladat megoldásához követni kell az utasításokat az adott sorrendben és nyomon kell követni az egyes lépések után, hogy a labdák milyen helyzetben vannak.

A robotkar kicseréli a labdákat. Mivel egyszerre csak egy labdát tud felvenni, félre kell tennie az első, hogy helyet csináljon a másodiknak. Ezért három helyre van szüksége a cseréhez.

Egyes programozási nyelvek esetében két „a”, „b” változó értékének cseréjéhez az első változó értékét ideiglenesen egy harmadik „c” változóban kell tárolnunk:

```
c = a
a = b
b = c
```

Ha az értékek egész számok, akkor van egy módszer arra, hogy ne kelljen egy új, harmadik változót létrehozunk:

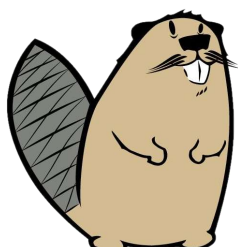
```
a = a + b
b = a - b
a = a - b
```

Néhány programozási nyelv, például a Python, támogatja a többszörös értékadást, úgynevezett értéksorozatok (tuple) segítségével, például így:

```
a, b = b, a
```

## WEBOLDAL

Algoritmus : <https://hu.wikipedia.org/wiki/Algoritmus>



## BÓLINTÁS SZÁMLÁLÁSA (2021-DE-05)

## SENIOR – NEHÉZ

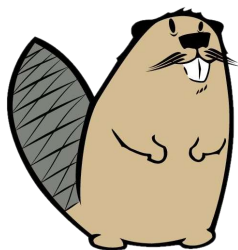
Hódvárosban az új jegyautomatákat úgy programozták be, hogy a fejmozdulatokra reagáljanak. A vevőnek annyiszor kell bólintania - lehajtania, majd ismét egyenesen tartani a fejét -, ahány jegyet szeretne vásárolni. Ezután fel kell emelnie a fejét és az automata kiadja a jegyeket.

A jegyautomata beépített kamerával rendelkezik, amely felismeri a vevő orrát, és folyamatosan méri az orr hosszát. A gép vezérlőprogramja az aktuális mérési eredményt Orrhossz néven menti, és a táblázat segítségével megkülönbözteti a vevők fejtartását:

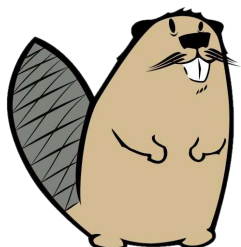
| Kamera mérése                                                                       | Orrhossz értéke | Fejtartás                        |
|-------------------------------------------------------------------------------------|-----------------|----------------------------------|
|    | 1               | A vevő egyenesen tartja a fejét. |
|   | 1,3             | A vevő lehajtotta a fejét.       |
|  | 0,7             | A vevő felemelte a fejét.        |

**Az alábbiak közül melyik vezérlőprogram valósítja meg helyesen a jegyautomata működését?**

|                                                                                                                                                                                                                                |                                                                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>A</p> <p>Számláló legyen 0</p> <p>ismételd amíg Orrhossz &lt; 0,8</p> <p>várj amíg Orrhossz &gt; 1,2</p> <p>várj amíg Orrhossz &lt; 1,1</p> <p>növeld meg a Számláló értékét 1-gyel</p> <p>adj ki Számláló darab jegyet</p> | <p>B</p> <p>Számláló legyen 0</p> <p>ismételd amíg Orrhossz &gt; 0,8</p> <p>várj amíg Orrhossz &gt; 1,2</p> <p>várj amíg Orrhossz &lt; 1,1</p> <p>növeld meg a Számláló értékét 1-gyel</p> <p>adj ki Számláló darab jegyet</p> |
| C                                                                                                                                                                                                                              | D                                                                                                                                                                                                                              |



|                                        |                                        |
|----------------------------------------|----------------------------------------|
| <b>Számláló legyen 0</b>               | <b>Számláló legyen 0</b>               |
| ismételd amíg <b>Orrhossz &lt; 0,8</b> | ismételd amíg <b>Orrhossz &lt; 0,8</b> |
| várj amíg <b>Orrhossz &lt; 1,1</b>     | várj amíg <b>Orrhossz &lt; 1,1</b>     |
| várj amíg <b>Orrhossz &gt; 0</b>       | várj amíg <b>Orrhossz &gt; 1,2</b>     |
| növekd meg a Számláló értékét 1-gyel   | növekd meg a Számláló értékét 1-gyel   |
| adj ki Számláló darab jegyet           | adj ki Számláló darab jegyet           |



## A helyes válasz az A)

A program felépítése előre meghatározott: Van egy ismétléses utasítás (melyet gyakran "ciklusnak" is nevezünk). Az ebben a ciklusban a megismételt utasítások közül az utolsó növeli a kiadandó jegyek számát. A két „várj” utasítással kell érzékelni a vevő bólintását: vagyis a vevő először lehajtja a fejét, majd ismét egyenesen előre néz (Ezért helytelen a D válasz, mert ott először tarja egyenesen és aztán hajtja le a fejét). Ezért az Orrhossz-ban tárolt értéknek először 1,3 körül kell lennie, majd ismét 1-hez közel. Ez megfelel az "Orrhossz>1,2" és utána az "Orrhossz<1,1" feltételeknek.

A ciklus utasításait addig ismételjük, amíg a vevő fel nem emeli a fejét: azaz olyan értéket mérünk, amely lényegesen kisebb, mint 1. Az egyetlen megfelelő feltétel a lehetőségek közül a <0,8.

Lehet, hogy észrevetted, hogy a program nem a táblázatban szereplő pontos értékek elérését, egyenlőségét vizsgálja. A gyakorlatban nem tudunk folyamatosan mérni, hanem csak bizonyos gyakorisággal (például 25-ször másodpercenként). Előfordulhat, hogy például az előretétekintés pontos értékét (1) egyáltalán nem kapjuk meg, mert az egyik mérésnél például 0,95-et, majd a következőnél 1,03 -at mérünk. Ezért legtöbbször közelítő értékeket, és kisebb-nagyobb hasonlításokat használunk pontos egyenlőségek vizsgálata helyett.

### MIÉRT INFORMATIKA?

A gépi látás (más néven számítógépes látás) az informatika egyik kiemelt területe, amelyen jelenleg is intenzív kutatások folynak. Mind elméleti megfontolások, mind gyakorlati alkalmazások nagy jelentőséggel bírnak.

A gépi látás jelentős alkalmazása az, hogy a fogyatékkal élőknek jobb lehetőséget biztosítanak a környezetükkel való autonóm interakcióra. A fogyatékoság súlyosságától függően előfordulhat, hogy egy személy csak nagyon kevés izmot képes irányítani. A világhírű fizikus, Stephen Hawking (1942-2018) például az arcizmának mozgatásával irányította a beszédkimeneti programot, miután egy betegség miatt nagyrészt elvesztette uralmát a többi izma felett.

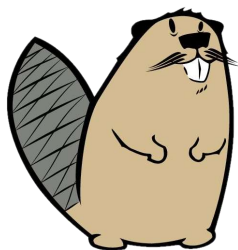
Példát azonban zenészeknél is lehet találni: ők általában mindkét kezüket használják hangszerük működtetésére és van olyan, amikor ez nem elég. Egyes zenészek, például az orgonisták a két kezükön kívül a lábukat is használják a játékhoz, és például egy egyszerű bólintással automatikusan lapozhatnak a kottában.

Ellentétben ennek a hódfeladatnak a vezérlőprogramjával, amely előre meghatározott értékeket használ a döntéseihez, a gépi látást gyakran kombinálják a gépi tanulóval. A programot ezután bizonyos gesztusokra képzik ki, sok példát és ellenpéldát mutatva minden egyes gesztusra. Így módon a szoftver statisztikai értékelést készít arról, hogy mit és hogyan kell értelmeznie. Magyarországon az ELTE IK Kutatói is több ilyen irányú fejlesztést és kutatást végeznek.

### WEBOLDALAK

[https://hu.wikipedia.org/wiki/G%C3%A9pi\\_tanul%C3%A1s](https://hu.wikipedia.org/wiki/G%C3%A9pi_tanul%C3%A1s)

[https://hu.wikipedia.org/wiki/Stephen\\_Hawking#Technikai\\_eszk%C3%B6z%C3%B6k\\_a\\_beteg\\_szoftverek\\_hasznalat%C3%A1ban](https://hu.wikipedia.org/wiki/Stephen_Hawking#Technikai_eszk%C3%B6z%C3%B6k_a_beteg_szoftverek_hasznalat%C3%A1ban)



## A TEKNŐS ÚTJA (2021-DE-07)

KISHÓD - NEHÉZ

BENJAMIN - KÖZEPES

KADÉT - KÖNNYŰ

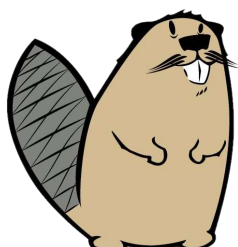
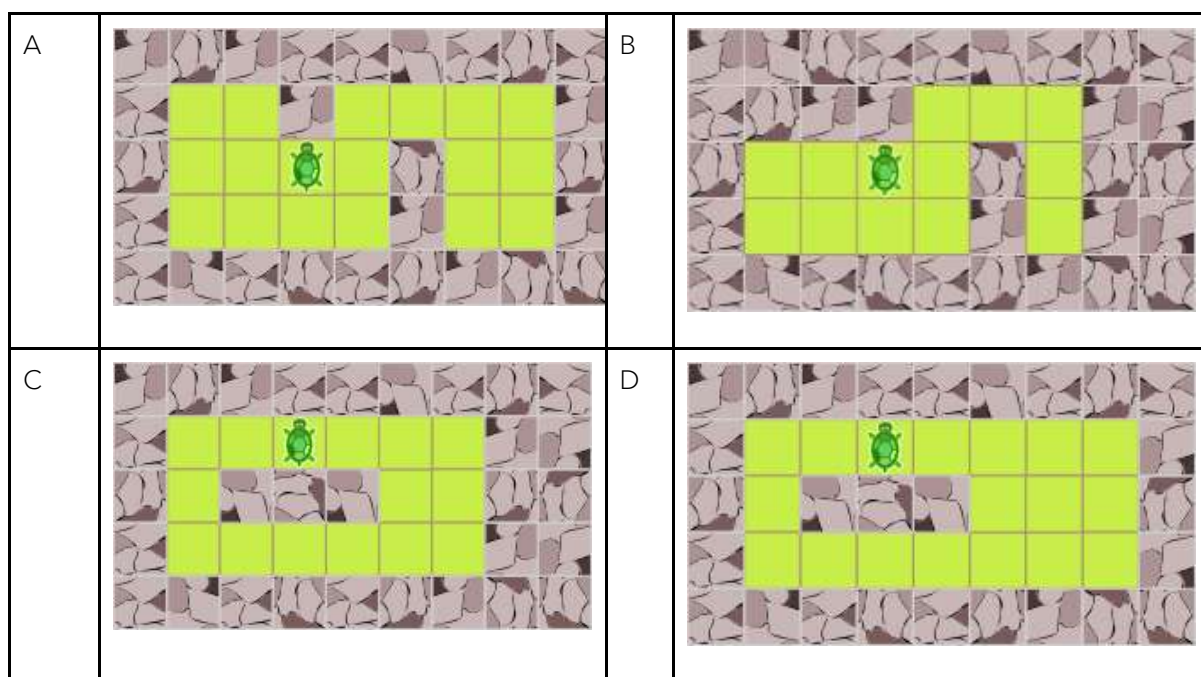
Egy teknősnek különböző kerteket kell lelegelnie. Minden kert négyzetekre van osztva, amelyeket vagy fű vagy kövek borítanak. A teknős nem léphet át a köves négyzetekre. Át tud azonban lépni az egyik füves négyzetről a másik, mellette található füves négyzetre.



A teknős mindegyik kertben arról a négyzetről indul, ahol a képen látható és pontosan egyszer léphet rá mindegyik füves négyzetre.

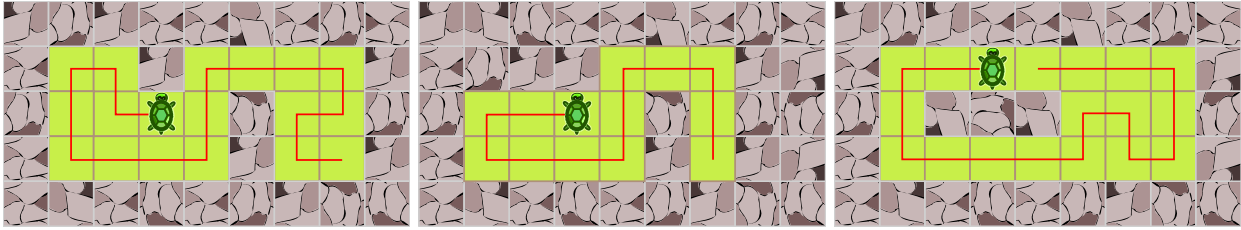
Sajnos így azonban **NEM** tudja teljesen lelegelni az egyik kertet (azaz nem tud minden füves négyzetet bejárni abban a kertben).

## Melyiket?



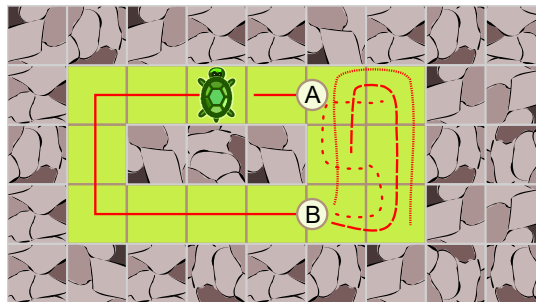
## A helyes válasz a C)

Az A, B és D kerteket a teknős teljesen legeltetheti.

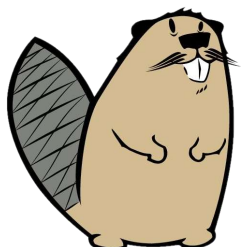
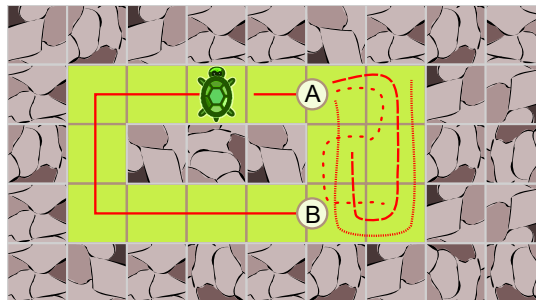


A teknős nem tudja a szabályok szerint lelegelni a C kertet. Csak 2 lehetősége van a kezdőpontjától elindulni:

- Ha először balra megy, a B ponthoz érkezik. Innen a jobb oldali 6 négyzetet kellene bejárnia, hogy a végén elérje az A pontot. De a B-ből származó lehetséges utak egyike sem ér véget A-nál.



- Ha a teknős először jobbra megy, akkor A-hoz érkezik, és le kellene legelnie a 6 jobboldali négyzetet, hogy a végén elérje a B pontot. Most az első esethez hasonlóan itt sem létezik ilyen lehetőség. Tehát még így sem létezik megfelelő út.

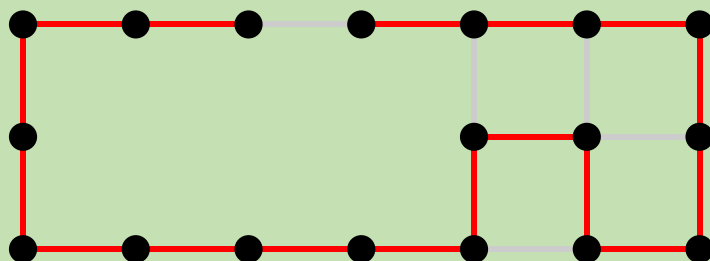


## MIÉRT INFORMATIKA?

Ebben a hód feladatban a teknősnek úgy kell megtalálnia egy utat a kertjében, hogy az minden négyzeten végigmenjen de pontosan csak egyszer. Ez az úgynevezett Hamilton-út problémája.

Az egyes kerteket (azaz a füves négyzeteket) a következőképpen tekinthetjük: Minden füves négyzet egy csomópont (pontként ábrázolva) és akkor vannak összekötve (élekkel), ha a teknős átmehet az egyikből a másikba.

A D kert tehát így néz ki:



Az ilyen struktúrák esetében (amelyeket az informatikusok és a matematikusok is gráfnak neveznek) tette fel magának a kérdést Sir William Rowan Hamilton a 19. században, hogy van-e olyan út az élek mentén, amely pontosan egyszer látogatja meg az egyes csomópontokat és meglátogatja az összeset. Az ilyen utat ezért Hamilton-útnak is nevezik.

Ha ugyanoda szeretnénk visszajutni, mint ahonnan elindultunk, akkor pedig a Hamilton-körrel beszélünk.

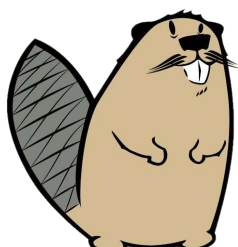
A kérdést, hogy létezik-e egyáltalán Hamilton-út, általában nagyon nehéz megoldani. Senki nem ismer olyan algoritmust, amely hatékonyan (többé-kevésbé rövid idő alatt) bármelyik gráf esetében eldöntené, hogy van-e Hamilton útvonal az adott gráfban vagy sem. Azt sem tudjuk, hogy létezik-e ilyen hatékony algoritmus. Ez tehát egy úgynevezett NP-teljes probléma, amelyek közül a Hamilton-út problémája az egyik leghíresebb.

(Több, bizonyos tulajdonságokkal rendelkező gráfban már megfogalmazták és bizonyították, hogy mikor létezik Hamilton-út, de ennek megtalálása nagyobb, összetett gráfok esetén igen költséges. A bizonyításokban olyan híres magyar matematikusok is részt vállaltak, mint Pósa Lajos és Erdős Pál)

## WEBOLDALAK

<https://hu.wikipedia.org/wiki/Hamilton-%C3%BAt>

<https://hu.wikipedia.org/wiki/Hamilton-k%C3%B6r>





## KEDVENC AJÁNDÉK\_A (2021-DE-08A)

KISHÓD - KÖZEPES

BENJAMIN - KÖNNYŰ

A hód család három ajándékot tartogat a három hód gyermek számára. Minden gyermek először megnevezi kedvenc ajándékát, majd a második kedvencet. Az ajándékokat a következő szempontok szerint szeretnék odaadni:

- A lehető legtöbb gyermek kapja meg a kedvenc ajándékát.
- A többiek pedig kapják meg a második kedvencet.



1: , 2: 



1: , 2: 



1: , 2: 



**Melyik ajándékot kapja a harmadik**

**hód gyermek?**

A) Bármelyiket, amelyiket szeretné.

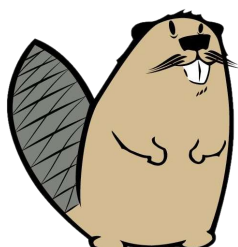


B) A kedvencét:



C) A második kedvencét:

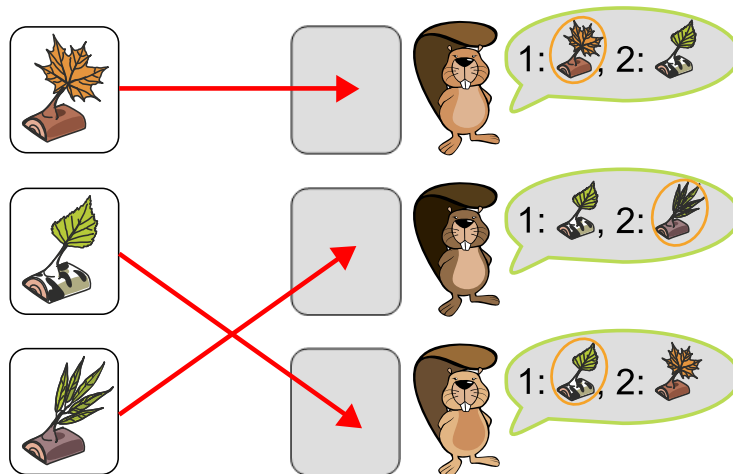
D) Nincs jó megoldás





## A helyes válasz a B)

A megfogalmazott szempontokat tekintve csupán egyetlen lehetséges ajándékosztás létezik:



Mivel csak a második hód szeretné a harmadik ajándékot (kép), ezért ezt fogja megkapni. Máskülönben valaki olyat kapna, ami nem a kedvenc ajándéka és nem a második kedvence. A másik két ajándék esetében a felosztás egyértelmű: mindenki megkaphatja kedvenc ajándékát.

## MIÉRT INFORMATIKA?

Ez a hódfeladat egy egyértelmű hozzárendelés-probléma: van két, ugyanannyi elemből álló halmazunk, melyek elemeit úgy szeretnénk párosítani (egymáshoz rendelni), hogy minden elemhez egy másik elem tartozzon. Itt ráadásul még a párosítások preferenciája, azaz "kívánság" - sorrendje is adott.

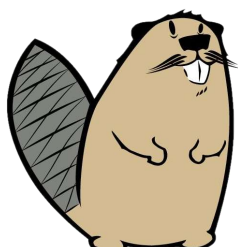
Az ilyen sorrenddel megadott hozzárendelési problémák nagyon bonyolulttá válhatnak. Az informatika segít nekünk az ilyen problémák mielőbbi megoldásában.

Az egyik lehetőség, hogy értéket adunk a kívánságoknak: például a kedvenc ajándék értéke 1, a második kedvencé pedig 2. Ekkor a célunk az, hogy minimálisra csökkentsük a teljes összeget, értéket. Az egyezés akkor optimális, ha nincs másik lehetőség, amely jobban teljesíti a hozzárendeléseket. Az informatikában az ilyen hozzárendelést rang-maximal-matching-nak nevezik. Sok egyeztetési probléma van. Az egyik például a "stabil házassági probléma". Érdekesen hangzik? Az informatika nagyon sokrétű tantárgy.

## WEBOLDALAK

[https://hu.wikipedia.org/wiki/Preferenci%C3%A1lis\\_szavaz%C3%A1s](https://hu.wikipedia.org/wiki/Preferenci%C3%A1lis_szavaz%C3%A1s)

[https://web.cs.elte.hu/blobs/diplomamunkak/bsc\\_matelem/2019/kelemen\\_adam\\_oliver.pdf](https://web.cs.elte.hu/blobs/diplomamunkak/bsc_matelem/2019/kelemen_adam_oliver.pdf)



## KEDVENC AJÁNDÉK\_B (2021-DE-08B)

KADÉT - KÖZEPES

JUNIOR – KÖNNYŰ

A hód család öt ajándékot tartogat az öt hód gyermek számára. Minden gyermek először megnevezi kedvenc ajándékát, majd a második kedvencet. Az ajándékokat a következő szempontok szerint szeretnék odaadni:

- A lehető legtöbb gyermek kapja meg a kedvenc ajándékát.
- A többiek pedig kapják meg a második kedvencet.



1: , 2:



1: , 2:



1: , 2:



1: , 2:



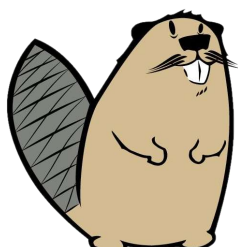
1: , 2:



Melyik ajándékot kapja a negyedik hód gyermek

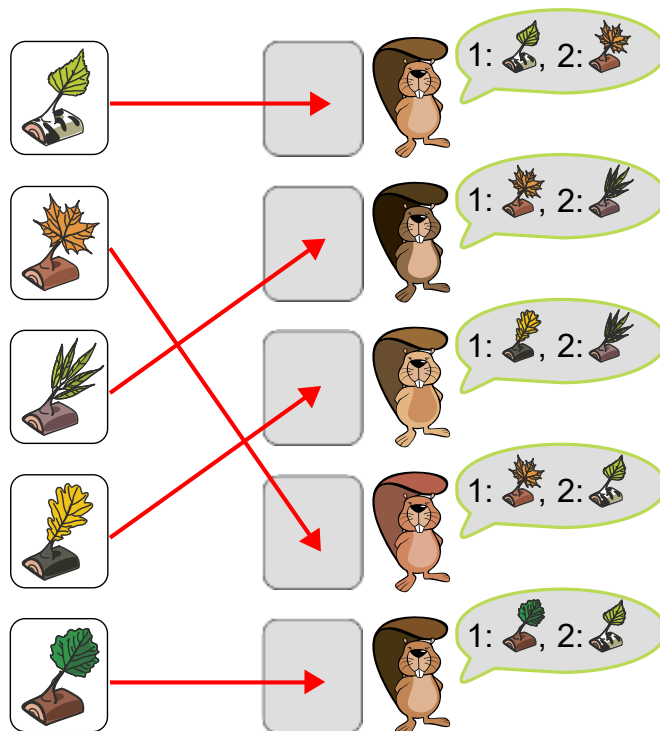
?

| A                              | B                | C                        | D                  |
|--------------------------------|------------------|--------------------------|--------------------|
| Bármelyiket a kedvencei közül. | A kedvencét:<br> | A második kedvencét:<br> | Nincs jó megoldás. |



## A helyes válasz a B)

A megfogalmazott szempontokat tekintve csupán egyetlen lehetséges ajándékosztás létezik:

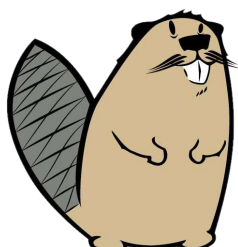


A fenti ábra négy hódhoz rendeli kedvenc ajándékát, egy hódhoz pedig második kedvenc ajándékát. Nem minden hód gyermek kaphatja meg kedvenc ajándékát, mert két hódnak ugyanaz a kedvence. Ezért nem lehet több hódot kijelölni kedvenc ajándékuk átvételére. Megjegyzés: Ha felülről lefelé végezzük a kiosztást, és a második ajándékot a második hódnak adjuk, akkor a negyedik hód egyik "kívánt" ajándékát sem kaphatja meg.

Az egyik megoldási stratégia az, hogy először rendeljük össze azokat az ajándékokat, amelyek csak egy gyermeknél szerepelnek (akár kedvencként, akár második kedvencként).

Ezzel a felülről harmadik és a felülről ötödik hód gyermekhez egyértelmű az ajándék. De ekkor a felülről második hódhoz is hozzá kell rendelnünk a felülről harmadik ajándékot, hiszen az már csak az ő kívánságai között szerepel.

Ezt követően már csak két gyermek maradt. Náluk egyértelműen látszik, hogy ugyanazok az ajándékok érdeklik őket, de a kívánságuk sorrendje (preferenciája) meghatározza kinek melyiket adjuk.



## MIÉRT INFORMATIKA?

Ez a hód feladat egy egyértelmű hozzárendelés-probléma: van két, ugyanannyi elemből álló halmazunk, melyek elemeit úgy szeretnénk párosítani (egymáshoz rendelni), hogy minden elemhez egy másik elem tartozzon. Itt ráadásul még a párosítások preferenciája, azaz "kívánság"-sorrendje is adott.

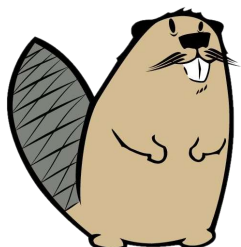
Az ilyen sorrenddel megadott hozzárendelési problémák nagyon bonyolulttá válhatnak. Az informatika segít nekünk az ilyen problémák mielőbbi megoldásában.

Az egyik lehetőség, hogy értéket adunk a kívánságoknak: például a kedvenc ajándék értéke 1, a második kedvencé pedig 2. Ekkor a célunk az, hogy minimálisra csökkentsük a teljes összeget, értéket. Az egyezés akkor optimális, ha nincs másik lehetőség, amely jobban teljesíti a hozzárendeléseket. Az informatikában az ilyen hozzárendelést rang-maximal-matching-nak nevezik. Sok egyeztetési probléma van. Az egyik például a "stabil házassági probléma". Érdekesen hangzik? Az informatika nagyon sokrétű tantárgy.

## WEBOLDALAK

[https://hu.wikipedia.org/wiki/Preferenci%C3%A1lis\\_szavaz%C3%A1s](https://hu.wikipedia.org/wiki/Preferenci%C3%A1lis_szavaz%C3%A1s)

[https://web.cs.elte.hu/blobs/diplomamunkak/bsc\\_matelem/2019/kelemen\\_adam\\_oliver.pdf](https://web.cs.elte.hu/blobs/diplomamunkak/bsc_matelem/2019/kelemen_adam_oliver.pdf)



## VIHARKÁR (2021-HU-02)

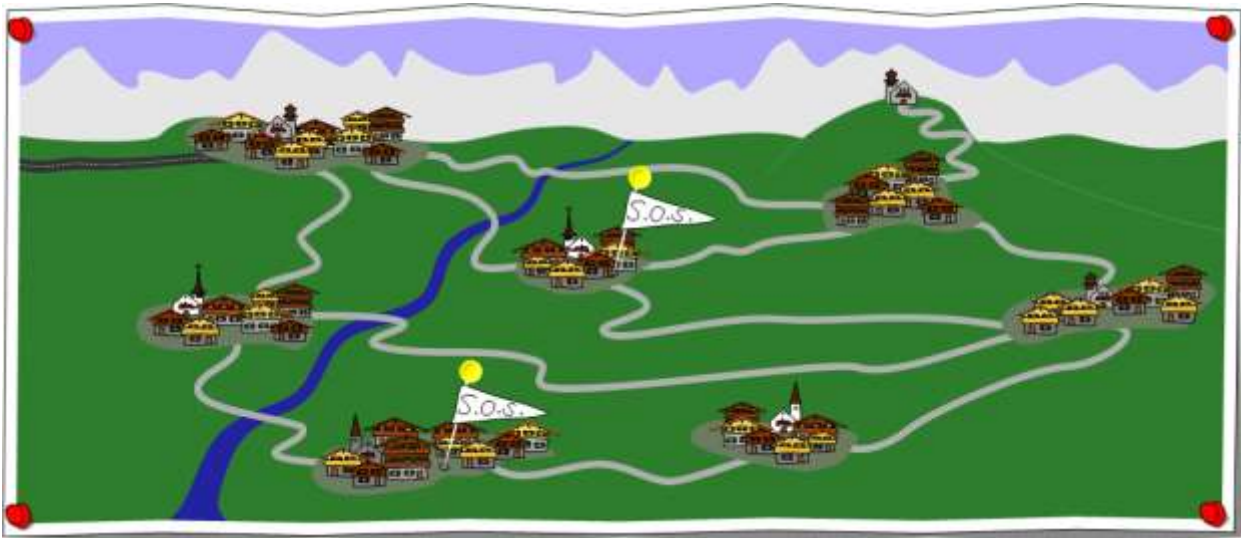
BENJAMIN - NEHÉZ

KADÉT - NEHÉZ

JUNIOR – KÖZEPES

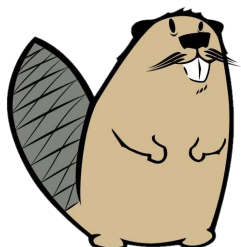
SENIOR – KÖNNYŰ

A nagyvárosból a képen látható úthálózatokon kapnak ellátást a falvak.



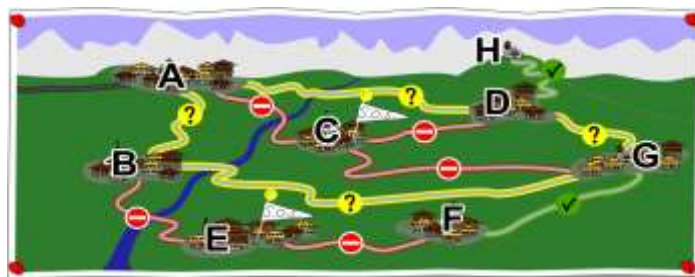
Egy nagyobb vihar után több falu arról számol be, hogy már nem lehet eljutni hozzájuk (lásd S.O.S jelzéseket), ebből arra következtethetünk, hogy egyes utak nem járhatóak.

**Legkevesebb hány út lett járhatatlan, ha azok a falvak, melyek nem jeleztek, biztosan elérhetőek?**



## A helyes válasz az 5

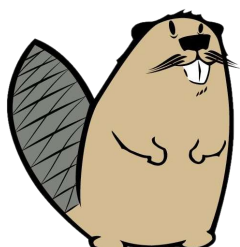
A képen bejelöltük, hogy mit tudunk az egyes utakról:



Kezdjük az elzárt falvakkal. Az E faluba vezető két út biztosan járhatatlan ( , különben az E faluba vagy B-ből vagy F-ből el lehetne jutni. Hasonlóképpen a C faluba vezető három út is járhatatlan (  ).

Ezután keressük azokat az utakat, amelyeknek járhatónak kell lenniük. A G és F falu közötti útnak járhatónak kell lennie ( , különben az F és E falu közötti elzárt út miatt az F falu nem lenne elérhető. A H templom és a D falu közötti útnak is járhatónak kell lennie ( , mivel H elérhető és csak D-n keresztül közelíthető meg.

Most az esetlegesen elzárt vagy járható utak maradnak (  ). Az, hogy nem egyértelmű, hogy ezek járhatóak-e vagy sem, annak az az oka, hogy az A, B, G és D falu egy körként vannak összekötve, ezért nem tudjuk megmondani, hogy melyik út járható vagy sem. Tehát a B falu elérhető az A falun keresztül, de a G falu is. Ugyanez vonatkozik a D falura is. A G falu vagy a B vagy a D falun keresztül elérhető. Az "A - B - G - D - A" körön lévő utak bármelyike (természetesen csak az egyik, hiszen nem jelöltek, azaz elérhetőek ezek a falvak) járhatatlanná válhatott, de ez a 4 falu továbbra is ellátható.



## MIÉRT INFORMATIKA?

Az úthálózatokhoz hasonlóan a számítógépes hálózatok kapcsolatai is meghibásodhatnak vagy pl. túlterheltekké válhatnak. A meghibásodások megelőzése érdekében gyakran terveznek biztonsági intézkedéseket, például több becsatlakozást, utat egy helyre. Ezt nevezik redundanciának.

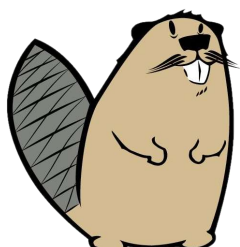
Egy rendszer hibáinak kijavítása olyan feladat, amellyel az informatikusoknak nagyon gyakran meg kell küzdeniük, nemcsak a számítógépes hálózatokban, hanem a szoftverfejlesztés során is. A hiba kijavításához meg kell határozni annak pontos forrását (helyét), és ez a folyamat általában fokozatos, többlépcsős folyamat. Néhány programozó úgy véli, hogy nincs hibátlan program, azaz soha nem találunk meg minden hibát, csak arra törekedhetünk, hogy minél többet megtaláljunk és kijavítsunk.

A programozókat sokszor kimondottan a hibakeresésre programozott alkalmazások is segítik.

## WEBOLDALAK

<https://hu.wikipedia.org/wiki/Redundancia>

<https://hu.wikipedia.org/wiki/Hibakeres%C5%91>



## FELHASZNÁLÓNÉV (2021-HU-03)

KISHÓD - KÖZEPES

BENJAMIN - KÖNNYŰ

Amikor egy új tanuló beiratkozik az iskolába, a következő módon kap felhasználónevet a számítógépes laborokhoz: veszik a diák vezetéknévének első három betűjét, majd a beiratkozás évének utolsó három számjegyét.

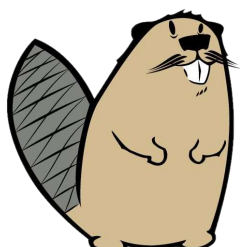
Tegnap 4 új diák iratkozott be az iskolába.

| Vezetéknév | Keresztnév |
|------------|------------|
| Vasöntő    | Karott     |
| Viharvész  | Eszmeralda |
| Vinkó      | Szilárd    |
| Beléndek   | Magrat     |

Ma egy újabb diák iratkozik be. A neve Vincellér Rozsdásbökő. Most kiderült, hogy a felhasználóneve ugyanaz, mint egy másik tanulóé.

**Kivel egyezik meg a felhasználóneve?**

- A) Vasöntő Karott
- B) Viharvész Eszmeralda
- C) Vinkó Szilárd
- D) Beléndek Magrat





## A helyes válasz a C)

Vincellér Rozsdásbökő a „vin021” felhasználónevet kapta: vezetéknéve Vincellér, melynek első három betűje „v”, „i” és „n”. 2021 -ben iratkozott be, tehát az év utolsó három számjegye 0, 2 és 1. Ha mindezt összeadjuk, az „vin021” lesz.

A többi tanuló felhasználónevét ugyanígy elkészíthetjük:

| Vezetéknév | Keresztnév | Felhasználónév |
|------------|------------|----------------|
| Vasöntő    | Karott     | vas021         |
| Viharvész  | Eszmeralda | vih021         |
| Vinkó      | Szilárd    | vin021         |
| Beléndek   | Magrat     | bel021         |

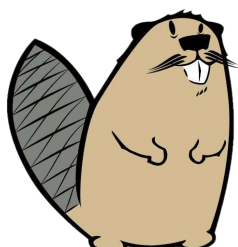
Ebből is kiolvasható, hogy Vinkó Szilárd és Vincellér Rozsdásbökő ugyanazzal a felhasználónévvel rendelkezne, hiszen ugyanabban az évben iratkoztak be.

## MIÉRT INFORMATIKA?

Ha egy számítógépet, erőforrást többen is megosztanak, fontos, hogy mindegyikük más felhasználónévvel jelentkezzen be. Ellenkező esetben ugyanazon felhasználónévvel rendelkező másik személy is elolvashatja vagy módosíthatja egy személy dokumentumait, e-mailjeit. Az iskola által ebben a feladatban használt rendszer ezért nem túl jó, mert a hasonló vezetéknévvel rendelkező diákok gyakran ugyanazt a felhasználónevet kapják.

Valójában a felhasználóneveket bonyolultabb szabályok alkalmazásával hozzák létre (például a keresztnév és vezetéknév használatával), de még akkor is előfordulhat, hogy két személy ugyanazt a felhasználónevet kapja. Ebben az esetben a vállalatok gyakran extra számot adnak a felhasználónév végéhez, hogy megkülönböztethessék őket.

Nem mindig biztonságos egy évet vagy dátumot megadni felhasználónevében (vagy jelszavában). Előfordulhat, hogy olyan információkat tár fel rólad, amelyeket inkább titokban tartanál.



## HÓDRALLY (2021-HU-04)

## SENIOR – NEHÉZ

Coraline és Tristan Hódrally társasjátékot játszanak.

Minden játékosnak van egy hódja () a táblán aminek kártyákkal irányítják a mozgását: például egy mezőt előre lépnek, balra vagy jobbra fordulnak (az aktuális irányhoz viszonyítva).

A játékosok kapnak 4 lapot, és elrendezik azokat abban a sorrendben, ahogy a hódot mozgatni akarják.

A hódok minden kártyalap esetében egymással egyszerre mozognak. Ha egyszerre próbálnak ugyanarra a mezőre lépni, akkor a kártyákon látható prioritási számok határozzák meg a sorrendet. Először a magasabb prioritási számú kártyán végrehajtott mozgás kerül végrehajtásra. Ha egy hód foglalt helyre lép, a másik hódot a mozgás irányában eltolja a következő mezőre.

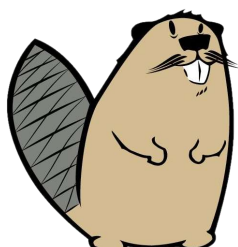


Itt láthatod, ahogy Coraline és Tristan elrendezték a lapjaikat:

|                                                                                                        |                      |                      |                      |                      |
|--------------------------------------------------------------------------------------------------------|----------------------|----------------------|----------------------|----------------------|
| <br>Coraline lapjai | <div>231<br/>↑</div> | <div>321<br/>↶</div> | <div>136<br/>↑</div> | <div>513<br/>↷</div> |
| <br>Tristan lapjai  | <div>123<br/>↑</div> | <div>543<br/>↷</div> | <div>564<br/>↑</div> | <div>631<br/>↶</div> |

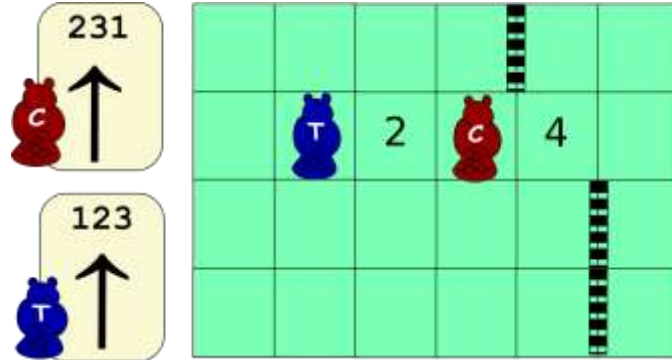
**Melyik számozott helyre kerül Coraline és Tristan hódja a négy kártya mozgásainak végrehajtása után, ha a következő helyeken és irányokba állnak?**

|  |                                                                                     |   |                                                                                     |   |  |
|--|-------------------------------------------------------------------------------------|---|-------------------------------------------------------------------------------------|---|--|
|  |                                                                                     |   |                                                                                     |   |  |
|  | 1                                                                                   | 2 | 3                                                                                   | 4 |  |
|  |  |   |  |   |  |
|  |                                                                                     |   |                                                                                     |   |  |

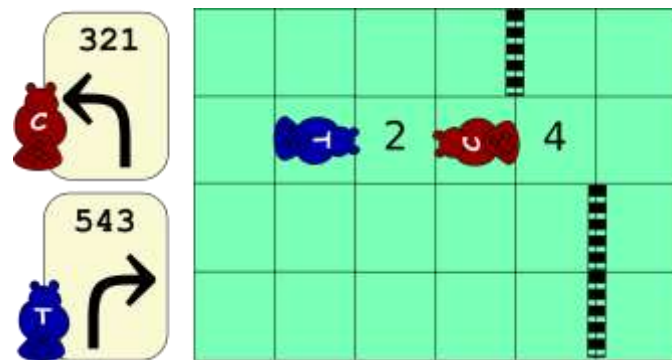


A helyes válasz: Tristané az 1-essel, Coraline hódja pedig a 2-essel jelölt helyre kerül

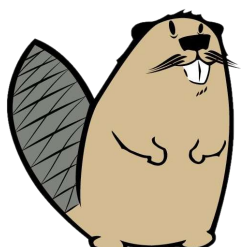
1. Az első kártyák mozgatai egyszerre végrehajthatóak: mindkét hód egy-egy mezőt lép előre (Coraline hódja a 3., Tristané pedig az 1. számozott mezőre kerül)

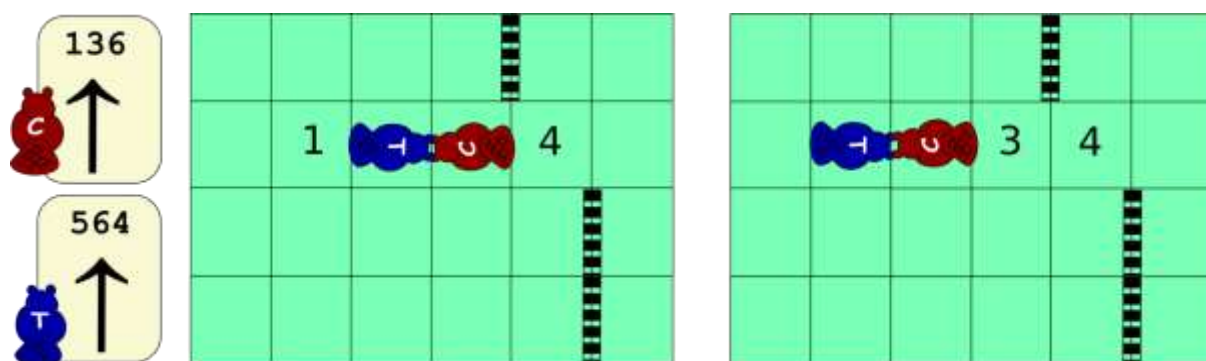


2. A második kártyák mozgatai is egyszerre végrehajthatóak, Coraline hódja balra, Tristané pedig jobbra fordul



3. A harmadik mozgatainál mindketten előre lépnének és ugyanarra a mezőre. Tehát a magasabb prioritási számú kártyát kell előbb végrehajtani, azaz Tristan hódja lép előbb: át a 2. számozott mezőre. Ezután lép Coraline, aki a "mozgásának irányában" eltolja Tristan hódját (hiszen azon a mezőn van, amire lépne). Így Tristan hódja visszakerül az 1. és Coralineé pedig át a 2. számozott mezőre.





4. Az utolsó lépésben mindkét hódnak már csak egy fordulást kell végrehajtania, azaz ezt is megtehetik egyszerre és nem lépnek másik mezőre.



### MIÉRT INFORMATIKA?

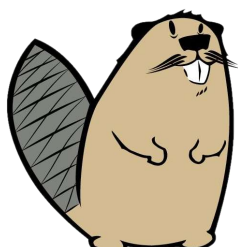
Egyes számítógépes programok, rendszerek egy-egy összetettebb feladatot kisebb részfeladatokra osztanak, és több folyamatot indítanak, hogy azok többnyire önállóan, egymással párhuzamosan dolgozzanak a kisebb feladatokon. Ezt párhuzamos végrehajtásának nevezik. Az egyidejű végrehajtás során előfordulhat, hogy egy folyamatnak várnia kell, hogy hozzáférjen egy megosztott erőforráshoz, vagy valamilyen módon befolyásolja egy másik folyamat eredménye. Fontos meghatározni a folyamatok viselkedésének koordinálására vonatkozó szabályokat ilyen helyzetekben.

Vannak olyan rendszerek, amelyek csak látszólag futtatnak párhuzamosan folyamatokat, igazából az erőforrások ütemezésének köszönhetően hol az egyik, hol a másik kap hozzáférést az adott erőforráshoz és "halad" a feldolgozásban.

A nagyobb számításoknál, folyamatoknál azonban érdemes valóban ki-, illetve szétosztani a részfeladatokat és azokat valóban - akár külön szervereken futtatva - feldolgozni.

### WEBOLDAL

[https://hu.wikipedia.org/wiki/P%C3%A1rhuzamos\\_sz%C3%A1m%C3%ADt%C3%A1stechnika](https://hu.wikipedia.org/wiki/P%C3%A1rhuzamos_sz%C3%A1m%C3%ADt%C3%A1stechnika)



## NYOMDA (2021-HU-05C)

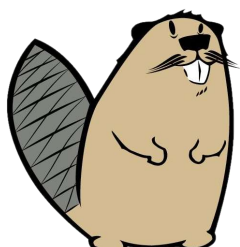
## KISHÓD - KÖNNYŰ

Gabinak négy különböző nyomdája van. Ezekkel készített egy képet úgy, hogy mindegyik nyomdát csak egyszer vette kézbe.



**Melyik nyomdát használta elsőként?**

| A)                                                                                  | B)                                                                                  | C)                                                                                   | D)                                                                                    |
|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
|  |  |  |  |

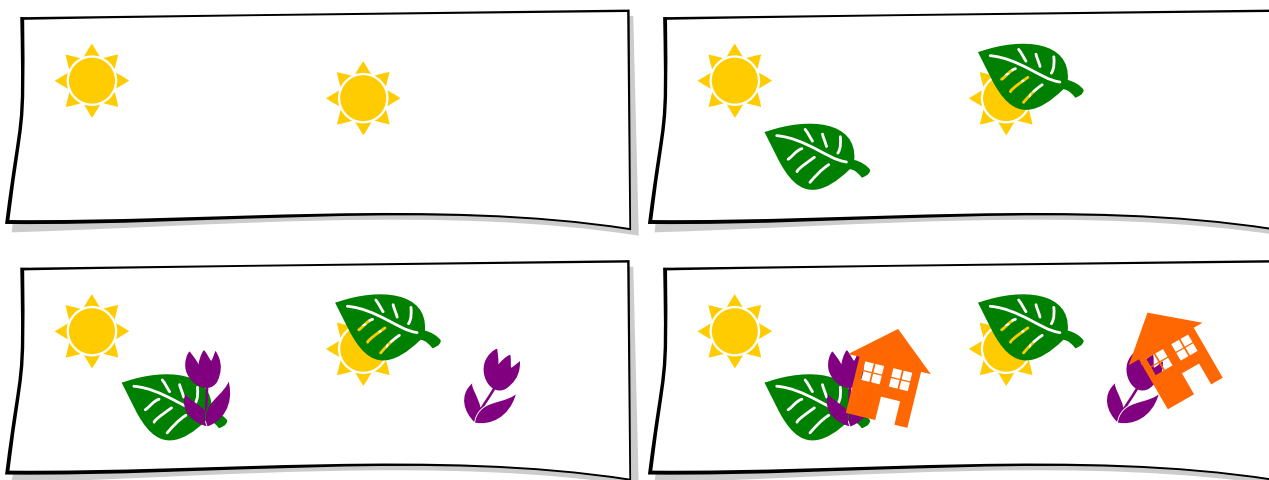


## A helyes válasz a C)



Az, hogy Gabi milyen sorrendben használta a nyomdákat, felismerhető abból, hogy melyik nyomdázott ábra melyik másik ábrát takarja.

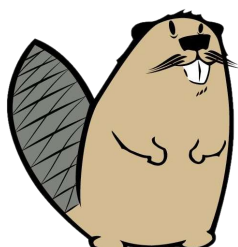
A nap nyomdát használta először, amit fed a levél. De a levelet fedi a virág, amit pedig a ház. Mivel minden nyomdát csak egyszer vett kézbe, ezért ez a sorrend egyértelmű.



## MIÉRT INFORMATIKA?

Gabi által használt módszer jól példázza a rétegek használatát. A rétegek segítségével egy kétdimenziós sík papíron, képen is kelthetjük a mélység és a háromdimenziós tér illúzióját.

A rajzfilmek, képek, animációk és számítógépes játékok készítésekor is érdemes több réteget használni nem csak az egymást fedő dolgok, objektumok megmutatására. A rétegek segítségével nem kell minden képkockán, jelenetben újra és újra ugyanazokat a dolgokat megrajzolni. A nem változó (mozgó) rétegek tartalma érintetlenül maradhat és akár a többi réteghez képest elfoglalt helyzetét is megváltoztathatja.

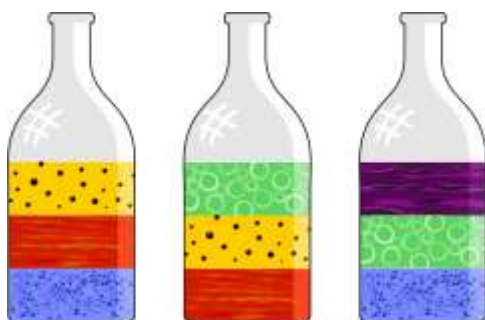


## SZÍNES FOLYADÉKOK (2021-ID-10)

KADÉT - NEHÉZ

JUNIOR – KÖZEPES

SENIOR – KÖNNYŰ



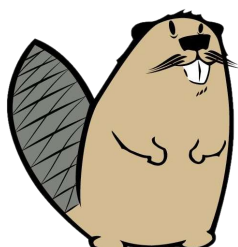
Gabinak 3 palackja van, amelyekbe rendre 3 féle folyadékot töltött. Az egyes folyadékok különböző színűek és különböző sűrűségűek.

Tudja, hogy az alacsonyabb sűrűségű folyadék a nagyobb sűrűségű folyadék felett van. Most szeretné látni, hogyan néz ki, ha minden színes folyadékot egy üvegbe tölt.

**Milyen sorrendben helyezkednek el a színes folyadékok egy üvegbe töltve?**



| A | B | C | D |
|---|---|---|---|
|   |   |   |   |



## A helyes válasz az A)

A képen az öt színes folyadék helyes elrendezése látható a nagy üvegben.



A sorrendet a következő stratégiát követve állapíthatjuk meg: Vegyük azt a folyadékot, amely nincs másik folyadék tetején, azaz nincs alatta folyadék. Ez kerüljön bele a nagy palackba és "vegyük ki" a kisebb palackokból.

Kezdetben az első és a harmadik palackban a kék folyadék legalul van, tehát nincs alatta egyik palackban sem másik folyadék. A piros folyadék a második palack alján található ugyan, de az első palackban a kék folyadék tetején van, és ezért kisebb sűrűségű, mint a kék. Tehát az első dolog, hogy vegyük a kék folyadékot, és ez kerüljön a nagy palack legaljára (valamint "vegyük ki" a kisebb palackokból). Most a vörös folyadék az egyetlen, amely nincs más folyadék tetején. Ha beleöntjük a nagy palackba, és "kivesszük" az első és a második palackból, akkor a sárga, a zöld és végül így folytatva a lila folyadék lesz az, amelynek sűrűsége éppen a legmagasabb, és amely alatt nincs más folyadék.

## MIÉRT INFORMATIKA?

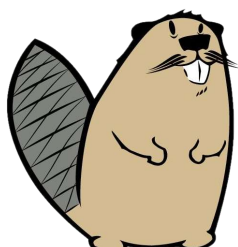
Egy anyagnak számos mérhető tulajdonsága van: forráspont, olvadáspont, elektromos vezetőképesség és sűrűség. Ebben az esetben a sűrűséget használtuk az anyagok sorba rendezéséhez kritériumként.

Az adatok rendezése számos számítógépes programban fontos szerepet játszik és rengeteg algoritmust ismerünk, melyek különbözőképpen rendeznek adatokat. Hogy mikor melyiket érdemes használnunk, az nagyban függ attól is, hogy mi alapján és milyen objektumokat szeretnénk rendezni.

Ebben a feladatban a folyékony rétegek sorrendjének meghatározására használt módszer topológiai rendezésnek nevezik. Olyan objektumok rendezésére szolgál, amelyek előd / utód kapcsolatai ismertek.

## WEBOLDAL

[https://hu.wikipedia.org/wiki/Topologikus\\_sorrend](https://hu.wikipedia.org/wiki/Topologikus_sorrend)





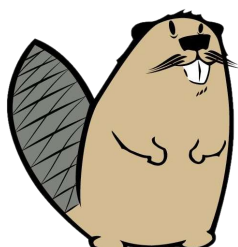
## MEZ (2021-IE-04)

KISHÓD - KÖNNYŰ

Anna ma világos ujjú, fekete galléros, de csíkozás nélküli mezt szeretne felvenni.

**Melyik mezt válassza?**

| A                                                                                 | B                                                                                 | C                                                                                  | D                                                                                   |
|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
|  |  |  |  |



## A helyes válasz a B)

Az A és a D mez ujjá fekete, azaz nem világos, így mára nem jó választás. (Az A ráadásul fehér galléros is).

A C mez csíkos, tehát ez sem megfelelő.

A B mez tökéletes mára: világos az ujjá, fekete a gallérja és nem csíkos.

### MIÉRT IINFORMATIKA?

Ebben a hódfeladatban dolgok közül kell kiválasztani azt, ami megfelel bizonyos feltételeknek, vagy éppen hogy nem felel meg.

Itt számos alfeltételt határoztak meg, mint például az ujjak színét, a mintát, és mindezeknek egyszerre (összekötve) kell teljesülniük. Az informatikusok logikai operátorokat (műveleteket) használnak ezekhez a kapcsolatokhoz.

Ha minden részfeltételnek egyszerre kell teljesülnie, akkor az ÉS (AND) operátort kell használni: az ujj színe = világos ÉS a gallér színe = fekete. Ha a több részfeltétel közül csak legalább az egyiknek kell teljesülnie, akkor a VAGY (OR) operátor használható. Ha egy részfeltétel nem szabad, hogy teljesüljön, akkor a NEM (NOT) operátor használható (pl. NEM (minta = csíkok))

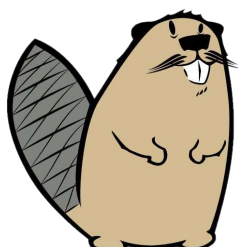
A lekérdezési nyelvek nagyon bonyolult feltételek megfogalmazására is használhatók az adatbázisokban való kereséshez.

Magukat a feltételeket azonban egyértelműen meg kell határozni. Például a "világos" feltétel nem olyan egyszerű, mint a "csíkos minta" feltétel. Ilyen esetben az informatikusok olyan programot vagy függvényt írnak, amely ellenőrzi, hogy a szín világos-e vagy sem. Ehhez az informatikusoknak világosan meg kell határozniuk, hogy pontosan mikor "világos" egy szín, különben nem lehet működő programot írni.

### WEBOLDALAK

[https://hu.wikipedia.org/wiki/Boole-algebra\\_\(informatika\)](https://hu.wikipedia.org/wiki/Boole-algebra_(informatika))

[https://hu.wikipedia.org/wiki/Logikai\\_m%C5%B1velet](https://hu.wikipedia.org/wiki/Logikai_m%C5%B1velet)



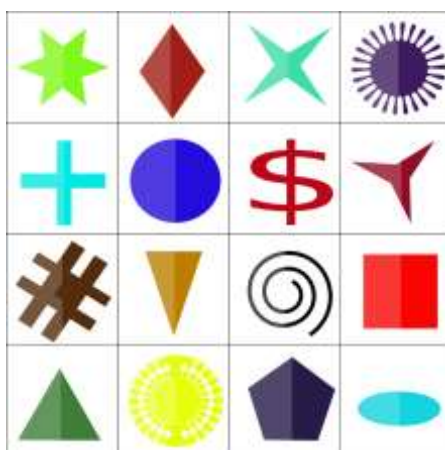
## FORMABONTÓ (2021-IN-05)

BENJAMIN - NEHÉZ

KADÉT - KÖZEPES

JUNIOR – KÖNNYŰ

Hód Helga Formabontót játszott. A játéktábla négyzetekre van osztva és Helga kezdetben a következőképpen helyezte el a formákat a táblán:



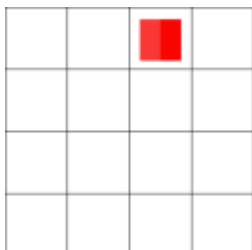
Ezután néhány formát felcserélt egymással.

Négy cserét hajtott végre, ebben a sorrendben:

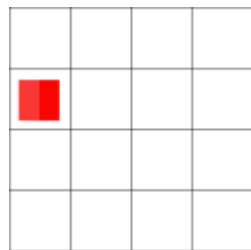
| 1                                                                                                                                                                         | 2                                                                                                                                                                         | 3                                                                                                                                                                           | 4                                                                                                                                                                             |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  ↔  |  ↔  |  ↔  |  ↔  |

Melyik négyzetbe került a  forma az utolsó cserét követően?

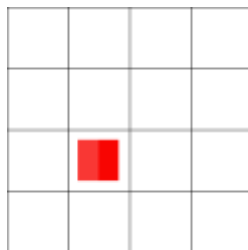
A



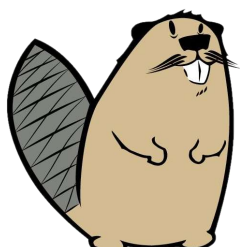
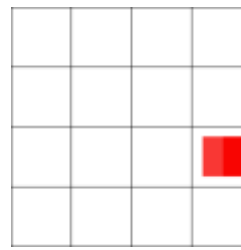
B



C

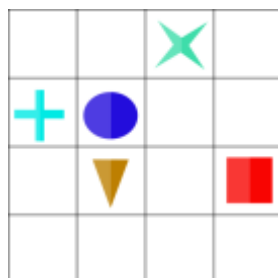


D

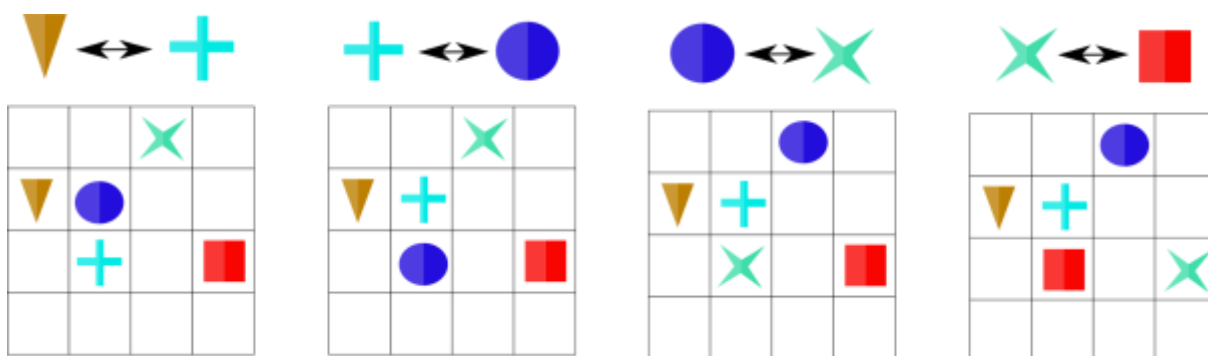


## A helyes válasz a C)

A feladat csak öt alakzat cseréjét foglalja magában. Így figyelmen kívül hagyhatjuk a nem érintett formákat. Az öt érintett alakzat kezdeti pozíciója az alábbiakban látható:



és a cserék után:



## MIÉRT INFORMATIKA?

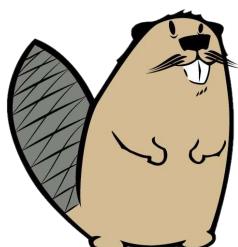
Ebben a hód feladatban a cserék olyan utasítások, amelyeket egymás után kell végrehajtani. Ez a legegyszerűbb algoritmikus elem, amelyet az informatikusok használni szoktak, amikor programot írnak. Szekvenciának is nevezik ezeket.

Amikor közel hasonló utasításokat szeretnének többször végrehajtani, akkor a programozók függvényeket írnak és használnak. Itt az egyes cserék között csak az a különbség, hogy melyik két formát cseréljük meg. Ezeket úgynevezett paraméterekként adhatjuk át egy előre megírt "csere" függvénynek, így ezek a részek, értékek rugalmasan használhatóak ugyanazoknak az utasításoknak az újabb használatánál.

## WEBOLDALAK

[https://hu.wikipedia.org/wiki/Utas%C3%ADt%C3%A1s\\_\(informatika\)](https://hu.wikipedia.org/wiki/Utas%C3%ADt%C3%A1s_(informatika))

[https://hu.wikipedia.org/wiki/F%C3%BCggv%C3%A9ny\\_\(programoz%C3%A1s\)](https://hu.wikipedia.org/wiki/F%C3%BCggv%C3%A9ny_(programoz%C3%A1s))



## ZSETONOSZLOPOK (2021-IT-01B)

JUNIOR – NEHÉZ

SENIOR – KÖZEPES

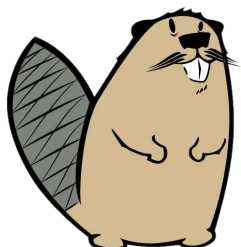
8 fehér és 8 piros zsetonunk van hét oszlopba szétosztva az alábbi ábra szerint:



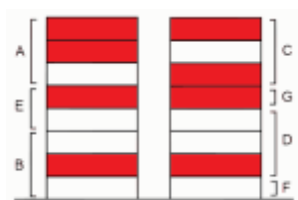
Ezeket az oszlopokat úgy szeretnénk egymásra helyezni (felosztás nélkül), hogy két egyforma oszlop alakuljon ki: a kapott két oszlop azonos magasságú legyen (nyolc zsetonból álljon mindkettő) és a színek alulról felfelé azonos sorrendben következzenek.

**Ha az (x, y, z,...) kifejezéssel jelöljük azt, hogy az x oszlopra ráhelyeztük az y-t, majd arra a z oszlopot, akkor az alábbi négy közül melyik NEM ilyen oszloppárt eredményez?**

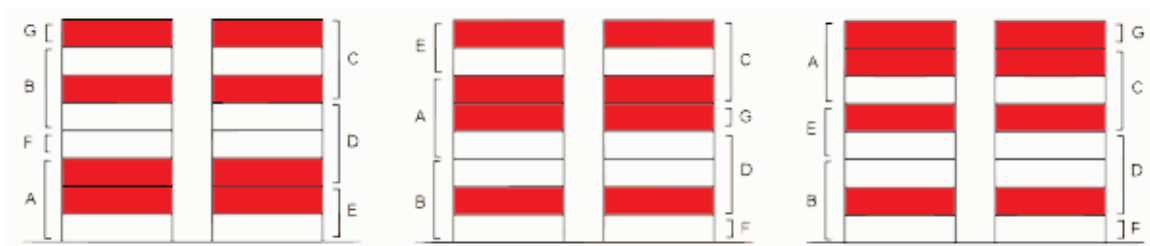
- A) (A, F, B, G) és (E, D, C)
- B) (B, A, E) és (F, D, G, C)
- C) (B, E, A) és (F, D, G, C)
- D) (B, E, A) és (F, D, C, G)



## A helyes válasz a C)



A többi válasz érvényes megoldás a feladatra; lásd az alábbi ábrát.



Mivel 16 zsetonunk van, amelyek közül 8 fehér és 8 piros, a két oszlop mindegyikének 4 fehér és 4 piros zsetonból kell állnia.

A négy A, B, C és D oszlop közül (3 zseton magas) kettőnek az egyik, a másik kettőnek a másik oszlopban kell lennie. De A és C (valamint B és D) nem lehetnek ugyanabban az oszlopban, különben lehetetlen lenne a színek kiegyensúlyozása (két fehér zsetont kellene hozzáadni az A-C párhoz, és két piros zsetont a B-D párhoz, de ilyen lehetőségünk nincs a maradék oszlopok között). Az is megmutatható, hogy A és D ugyanabba az oszlopba (és B és C a másikba) helyezése sem vezet eredményre.

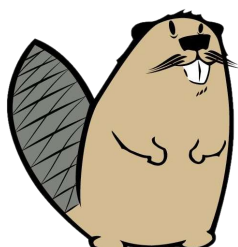
## MIÉRT INFORMATIKA?

Képzeld el, hogy van egy zsákod, sok érmével, és két egyenlő értékű részre szeretnéd osztani azokat. Ha páros számú és azonos értékű érméid lennének, akkor a probléma könnyen megoldható: elegendő két halomra osztani, amelyek azonos számú érmével rendelkeznek; de ha az érmék különböző értékűek, akkor a feladat kissé nehezebbé válik.

Az ilyen típusú problémákat partíciós problémának nevezzük, és a feladat annak eldöntése, hogy egy pozitív egész számokból álló multihalmaz (azaz egy halmaz, amely minden elemhez egy előfordulási gyakoriságot is rendelünk) felosztható-e két alhalmazra úgy, hogy az elsőbe kerülő számok összege egyenlő a második részhalmaz számainak összegével.

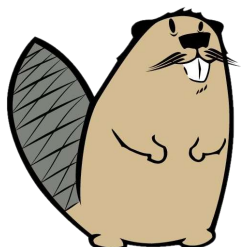
Ez a probléma (általában) NP-teljes (azaz a gyakorlatban nincs olyan ismert megoldás, amely minden esetben hatékony); de megoldható pseudo-polinomiális időalgoritmussal, dinamikus programozáson alapulva).

Egy hagyományos megoldási stratégia a "nyers erő módszere" (brute force), amely egy úgynevezett "teljeskörű tagolás" (exhaustive search) algoritmuson keresztül valósítható meg. Ennek lényege, hogy felsoroljuk az összes lehetséges részhalmazt és a feltételeink mellett elkezdjük ezeket vizsgálni.



Például a feladat esetében a hét oszlop összes részhalmaza, amelyek mindegyikében nyolc zseton szerepel:  $\{A, B, E\}$ ,  $\{A, B, F, G\}$ ,  $\{A, C, E\}$ ,  $\{A, C, F, G\}$ ,  $\{A, D, E\}$ ,  $\{A, D, F, G\}$ ; itt megállhatunk, hiszen folytatva a már felsorolt halmazok kiegészítő halmazait találjuk meg. Kihúzzhatjuk közülük az  $\{A, C, E\}$  és  $\{A, C, F, G\}$  halmazokat, mivel a hozzájuk tartozó oszlopokban összesen öt piros zseton található. Ezért csak négy halmaz maradt; mindegyik és a hozzájuk tartozó kiegészítő halmazok esetében végigpróbáljuk az összes lehetséges párosítást, és vizsgáljuk, hogy ezek közül bármelyik két azonos oszlopot eredményez-e. Esetünkben az  $\{A, B, E\}$  halmaz két megoldáshoz, míg az  $\{A, B, F, G\}$  halmaz csak egy megoldáshoz vezet, az utolsó két halmaz pedig egyhez sem. Jól látszik, hogy az eljáráshoz szükséges idő nagyon gyorsan fog nőni, ahogy az oszlopok kezdeti száma (esetünkben csak hét) növekszik.

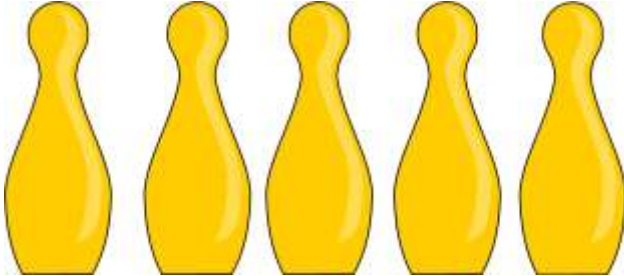
A partíciós (optimalizálási) problémák sokféle kontextusban fordulnak elő, mind a valós életben, mind az informatika fejlett területein, például a mesterséges intelligenciában és a gépi tanulásban.



## KAYLES (2021-IT-02B)

## SENIOR – NEHÉZ

Anna és Bendegúz sorban egymás mellé állítottak öt bowling bábút.

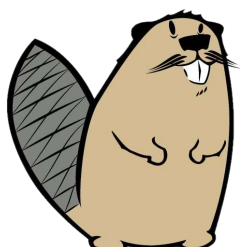


Két lehetőség közül választanak: vagy csak egy, vagy két szomszédos bábút dönthetnek el. Aki az utolsó bábút ledönti, az nyer. Anna és Bendegúz is jó játékosok, így mindig a lehető legjobb döntést hozzák.

### Hogyan kezdje el Anna a bábuk eldöntését, ha mindenképpen nyerni szeretne?

(Az eldöntött bábukat körrel jelöltük)

| A | B | C | D | E |
|---|---|---|---|---|
|   |   |   |   |   |





## A helyes válasz az E)

Nézzük meg a megoldásokat - és a lehetséges folytatásokat:

A) Ha Anna ledönti az első bábut, Bendegúz nyer azzal, hogy ledönti a „harmadik és negyedik bábu” párost. (Szimmetrikusan nézve ugyanez történik, ha Anna ledönti az ötödik bábut.)

B) Ha Anna leüti az első két bábut, Bendegúz nyer a negyedik bábu ledöntésével. (Ugyanez történik, ha Anna ledönti az utolsó két bábut.)

C) Ha Anna ledönti a második bábut, Bendegúz nyer azzal, hogy ledönti a „harmadik és negyedik bábu” párost. Hasonló helyzet áll elő, mint az A esetben. (Szimmetrikusan nézve ugyanez történik, ha Anna ledönti a negyedik bábut.)

D) Ha Anna leüti a „második és a harmadik bábu” párost, Bendegúz nyer a negyedik bábu ledöntésével. Hasonló helyzet áll elő, mint a B esetben.

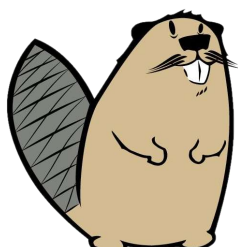
E) A harmadik bábu ledöntésével Anna két pár bábu osztotta a sort. Függetlenül attól, hogy Bendegúz egy vagy 2 bábut dönt le, Anna a következő lépésben ugyanannyit ledöntve a „másik” párosból nyer, illetve nyerő helyzetben marad.

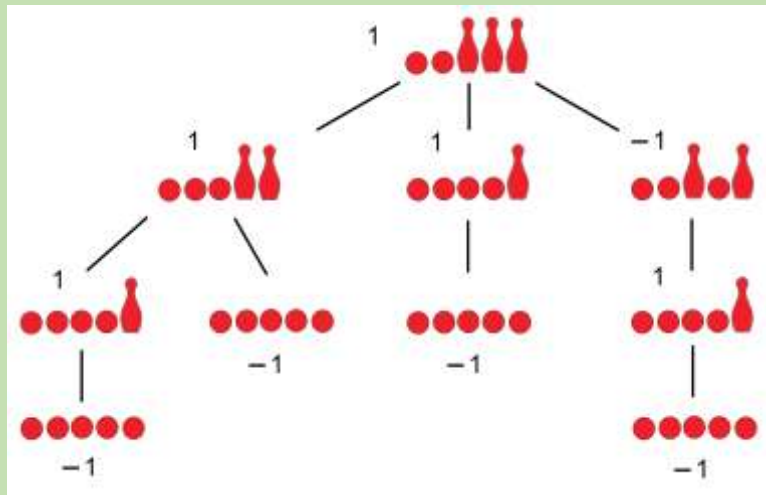
### MIÉRT INFORMATIKA?

A Kayles egy pártatlan (kombinatorikus) játék (azaz olyan játék, amelyet nem jellemez az „anyagi” megkülönböztetés és a véletlen lehetősége, és a játékosok felváltva lépnek). Henry E. Dudeney ihlette (The Canterbury Puzzles and Other Curious Problems, Heinemann, London 1907, Puzzle No. 73), aki 13 bábuval fogalmazta meg az első feladványát. A „kayles” név a francia „quilles” -ből származik, ami „bowlingot” jelent.

A játékról általánosságban elmondható, hogy egy  $n (> 0)$  bowling bábuból álló sorban a két játékos felváltva ledönt egy vagy két szomszédos bábut, amíg az összes bábu le nem dőlt. A normál játékszabály szerint az nyer, aki ledönti az utolsó bábut. Ebben az esetben az első játékos nyerő stratégiával rendelkezik bármely  $n (> 0)$  bábusorra: az első játékos biztosítja a győzelmét azzal, hogy az ellenfélnek két egyenlő (és leválasztott) bábusort hagy: ha  $n$  páratlan, az első játékos a középső bábut dönti le; ha  $n$  páros, akkor pedig a két középső bábut. Bármilyen akciót is hajt végre a második játékos a két sorozat egyikén a következő körökben, az első játékos „átmásolja” azt a másikra.

A Kayles egy speciális esete az úgynevezett „oktális játékoknak”, amelyek során objektumokat, tárgyakat távolítanak el egy halomból, esetleg sorokba, verembe vagy más formákba rendezve. Legtöbbször megalkotható egy nyerő stratégia, melyhez segítségül egy úgynevezett játékfát tudunk rajzolni: minden állás esetén megrajzoljuk, mi lehet a következő állás (most a szimmetrikus helyzeteket kihagytuk).





A legutolsó állapotból pedig visszakövethető, hogy ki nyer, milyen lépéseken keresztül.

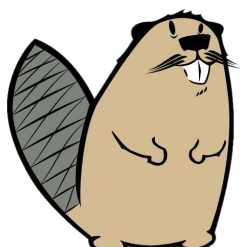
Ez az átgondolás, azaz egy fa vagy gráf felrajzolása arra, hogy melyik lépésből hova juthatunk el, majd egy olyan út (bejárás) megkeresése, ami a legmegfelelőbb, sok számítógépes játéknál a gépi oldal által használt megoldás. Ez azonban időnként nagyon nagy fát, gráfot eredményez (gondolj bele pl. a sakk első pár lépésének lehetőségeibe). Ezért előfordul, hogy a számítógép csak részfákat épít fel (az adott lépéstől 4-5 lehetséges mélységig), illetve ha használnak gépi tanulást, akkor a már "megismert" rossz utakat is ki tudja zárni.

#### WEBOLDALAK

<https://hu.wikipedia.org/wiki/J%C3%A1t%C3%A9kelm%C3%A9let>

<https://en.wikipedia.org/wiki/Kayles>

<https://web.cs.elte.hu/~veghal/orak/jatekelm-kombin.pdf>



## GOLYÓK (2021-KR-02)

BENJAMIN - KÖZEPES

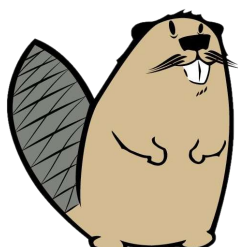
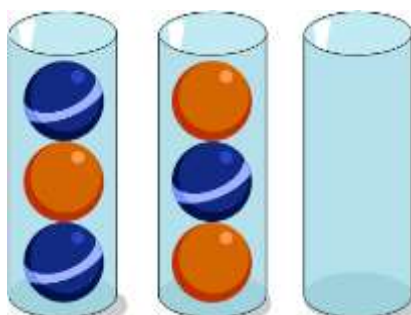
KADÉT - KÖNNYŰ

A hódok golyókat szeretnének a színük szerint szétválogatni úgy, hogy végül minden golyó két pohárban legyen és a golyóknak poharanként azonos színűnek kell lenniük. A válogatás során ezt a három szabályt is követik:

- ➡ 1. szabály: Egy lépésben egyszerre csak egy pohár legfelső golyóját lehet mozgatni.
- ➡ 2. szabály: A golyó áthelyezhető egy üres pohárba.
- ➡ 3. szabály: A golyó áthelyezhető egy pohárba, ha van még szabad hely és az alatta lévő golyó azonos színű.

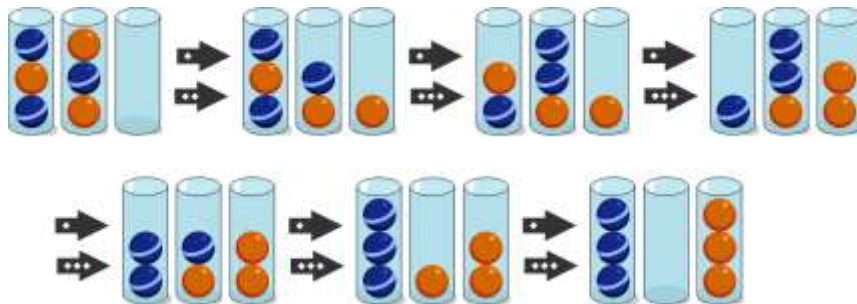


**A szabályokat követve legkevesebb hány lépésben tudják ezt a hat golyót színük szerint szétválogatni?**



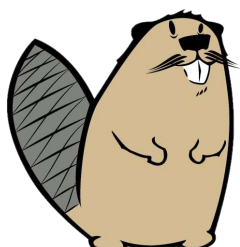
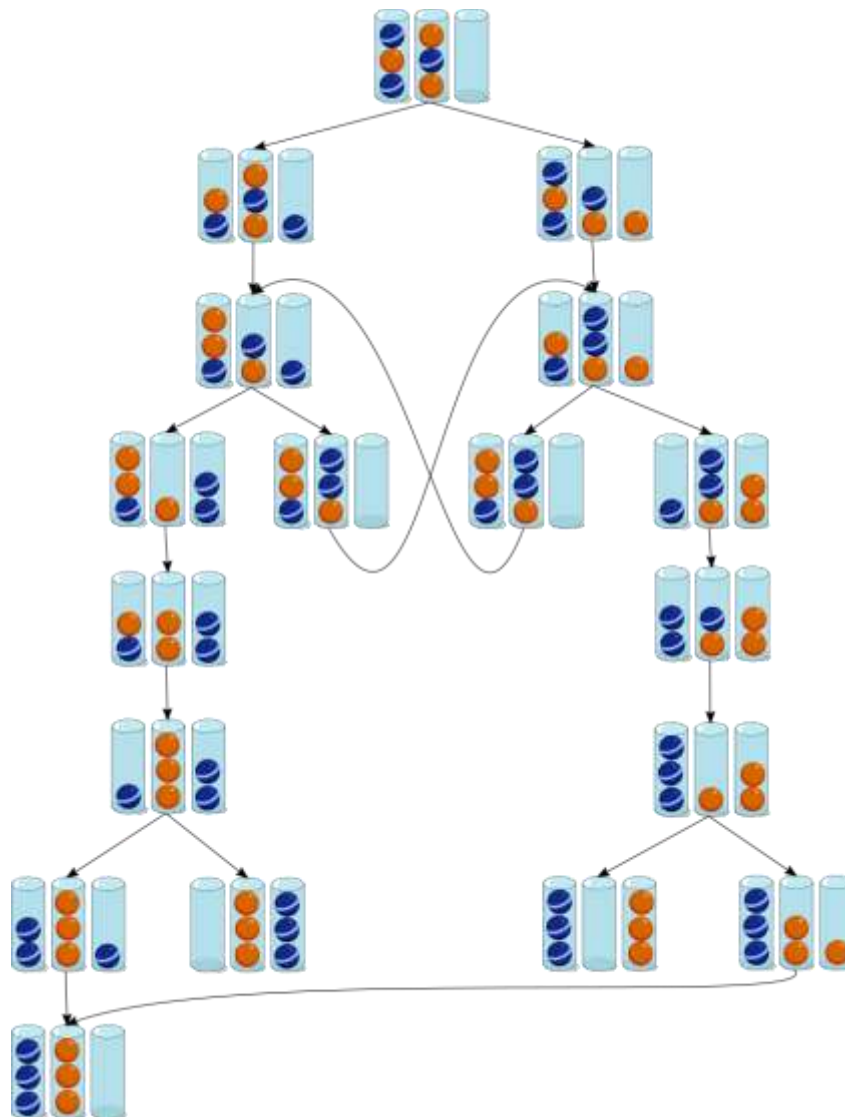
## A helyes válasz: 6 lépésben

A golyók szétválogatásához legalább 6 lépésre van szükség. Egy lehetséges megoldás:



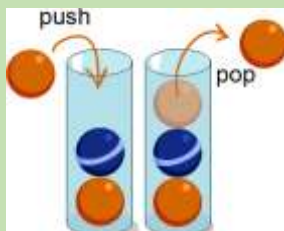
Természetesen más megoldások is léteznek, de ebből a kezdőállapotból nem oldható meg a feladat kevesebb lépésben.

Ehhez akár fel is rajzolhatjuk a lehetséges lépéseket (a visszalépéseket nem jelölve):



## MIÉRT INFORMATIKA?

Ebben a feladatban a golyókat hasonló módon mozgatjuk, mint ahogyan a számítógép kezeli a veremben (egy gyakran használt adatszerkezetben) lévő adatokat: csak egy elemet (a feladatban a golyókat) adhat hozzá a tetejéhez (push), és csak a felső elemet (pop) távolíthatja el.



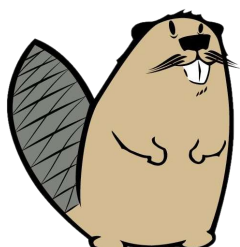
A számítógép csak akkor fér hozzá a lejjebb lévő elemekhez, ha a fentit eltávolítják. És a számítógép mindig a legutoljára mentett elemet távolítja el először. Az informatikusok ezt az elvet az "utolsó az első"-elvnek nevezik (röviden LIFO).

A megoldáshoz használt átgondolás, azaz egy fa vagy gráf felrajzolása arra, hogy melyik lépésből hova juthatunk el, majd egy olyan út (bejárás) megkeresése, ami a legrövidebb, sok számítógépes játéknál a gépi oldal által használt megoldás. Ez azonban időnként nagyon nagy fát, gráfot eredményez (gondolj bele pl. a sakk első pár lépésének lehetőségeibe). Ezért előfordul, hogy a számítógép csak részfákat épít fel (az adott lépéstől 4-5 lehetséges mélységig), illetve ha használnak gépi tanulást, akkor a már "megismert" rossz utakat is ki tudja zárni.

## WEBOLDALAK

[https://hu.wikipedia.org/wiki/Verem\\_\(adatszerkezet\)](https://hu.wikipedia.org/wiki/Verem_(adatszerkezet))

<https://hu.wikipedia.org/wiki/J%C3%A1t%C3%A9kelm%C3%A9let>



## HIBAKERESÉS (2021-LT-07)

BENJAMIN - NEHÉZ

KADÉT - KÖZEPES

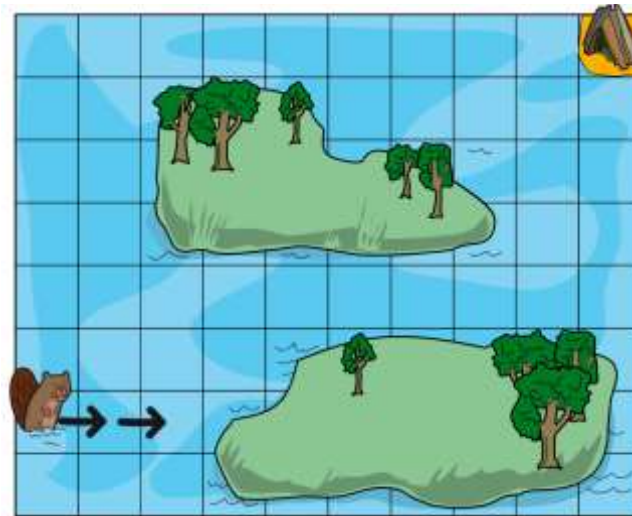
JUNIOR – KÖNNYŰ

Zsófi egy olyan programot írt, ami úgy segít hazajutni Hód Hubának, hogy nem kell elhagynia a vizet. Tehát csak úsznia kell, gyalogolnia nem.

A program az alábbi 7 utasításból áll:

2→ 2↑ 5→ 4↑ 1→

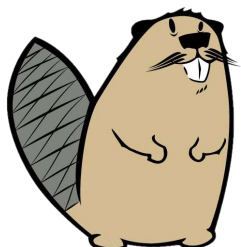
Az első utasítás biztosan helyes, és Huba előre úszik a képen látható módon:



A következő négy utasítás valamelyike azonban hibás!

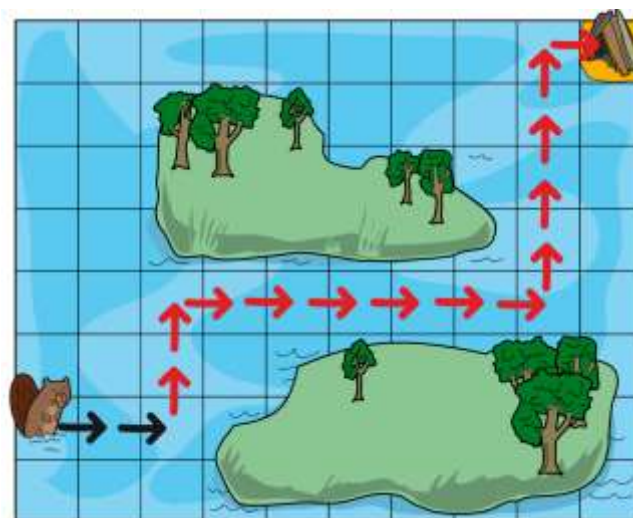
**Melyik az az egy utasítás, amit megváltoztatva már biztosan hazajuthat Huba (anélkül, hogy el kellene hagynia a vizet)?**

- A) 2 ↑
- B) 5 →
- C) 4 ↑
- D) 1 →

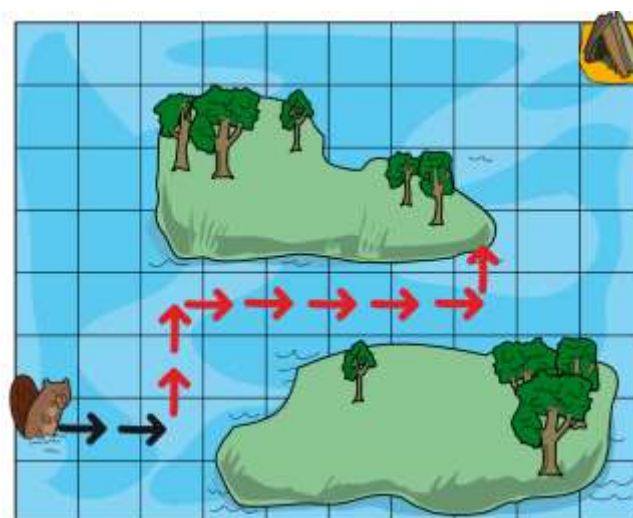


## A helyes válasz a B)

Az 5→ utasítást 6→ utasításra kell módosítani. Így Huba pontosan eléri azt a mezőt, ahol a háza van, ahogy ez látható is az alábbi képen:



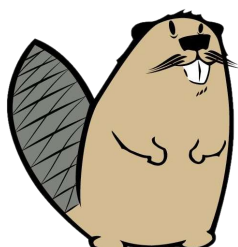
Ha Huba megpróbálja követni Zsófi eredeti programját az 5→ utasítás után, amikor elindul felfele (4↑), egy szigetre ér:



Tehát el kellene hagynia a vizet.

Ha eggyel tovább úszik, akkor a többi utasításban már semmi változtatásra sincs szükség, Huba gond nélkül hazaér.

Amennyiben nem lenne megkötés, hogy Hubának a vízben kell maradnia, bármelyik jobbra lépő utasításnál megváltoztathatnánk a számot (megnövelhetnénk eggyel), hiszen az út hosszából már csak egy "lépés" hiányzik ebbe az irányba.





## MIÉRT INFORMATIKA?

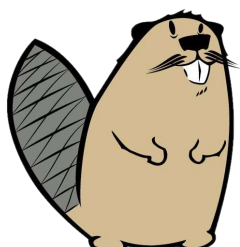
Egy program olyan utasításokból áll, amelyeket egy adott probléma megoldására használnak és olyan műveletek és döntések sorozatát képviseli, amelyeket meg kell hozni a probléma megoldása érdekében. Esetünkben egy adott ponttól kezdve egy másik pontba való eljutás lesz a probléma megoldása.

Ebben a feladatban egy hibát kell megkeresnünk, mely felmerült a probléma megoldása során. Ezt a folyamatot hibakeresésnek (debugging) nevezik, ami egy fontos készség a programozók számára.

ezt kellene kifejtteni.... mint a HU-02-esben.... Bence? megcsinálod?

## WEBOLDAL

<https://hu.wikipedia.org/wiki/Hibakeres%C5%91>





## HÓDIA KLÁNJAI (2021-PH-03)

JUNIOR – NEHÉZ

SENIOR – KÖZEPES

Hódiában öt, egykor hadakozó klán él: Bináriánok, Smartfoniánok, Checkboxiánok, Hexadecik és Laptopik. Mindegyik házban egyértelműen jelzik, melyik klánhoz tartoznak, amint az a képen is látható:

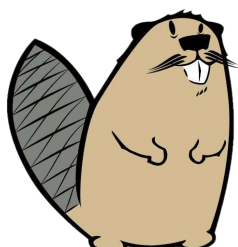


Mivel régóta békésen élnek egymás mellett, elhatározzák, hogy egyesítik a klánokat. Ehhez a következő szabályokat fektetik le:

- Egyszerre csak 2 klán egyesülhet.
- Az egyesülő klánok házainak mindegyikében egy hetes ünnepséget rendeznek. Az egyesülési ünnepség időtartama hetekben tehát megegyezik a két egyesülő klán házainak számával.
- Ezen idő elteltével a két klán már csak egy klán. Ekkor folytatható a klánok egyesítése.

A klánok úgy döntenek, hogy a lehető legrövidebb időn belül végrehajtják az egyesülést. Ezt csak az egyesülések sorrendjének gondos megtervezésével lehet megtenni.

**Legkevesebb hány hétre van szükség ahhoz, hogy minden klán egyesüljön?**



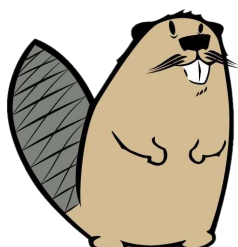
## A helyes válasz: 33 napra

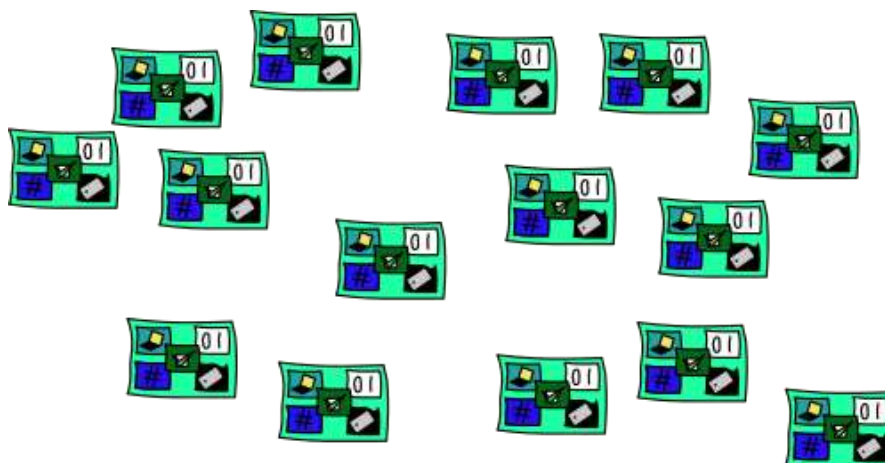
Az optimális stratégia az összes klán egyesítéséhez szükséges hetek minimalizálására az, hogy minimalizáljuk az egyes egyesülésekben résztvevő házak számát.

Az elsőnek egyesített klánok házait a további egyesülésekben figyelembe kell venni. Ezért van értelme először egyesíteni a legkisebb klánokat, hogy a nagyobbaknak ne kelljen túl gyakran részt venniük az ünnepségeken. Ennek eléréséhez a két legkevesebb házzal rendelkező klánt kell egyesíteni minden lépésben.

Ezt a lépésenkénti folyamatot a következő táblázat mutatja be, az egyértelműség kedvéért elhagytuk a klán neveket, és csak a klán méreteit tettük zárójelbe:

|                                                                                                  |                                                                                      |
|--------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| Az (1) és a (2) házból álló klánok egyesítésével egy új klán (3) jön létre. Ez három hétig tart. |    |
| A (3) és (3) nagyságú klánokat hat hét alatt egyesíthetik (6).                                   |   |
| Ezután (4) és (5) házból álló klánok kilenc hét alatt egyesülhetnek (9).                         |  |
| Végül a maradék (6) és (9) nagyságú klánok kerülhetnek összevonásra. Ez 15 hetet vesz igénybe.   |  |





Azaz az egész egyesítés  $15+9+6+3=33$  hét alatt lezárható.

Az itt megadott stratégia nem az egyetlen, amely optimális megoldáshoz vezet. Kezdetben a (4) és (5) is egyesíthető, és csak ezután következik az (1) és (2) egyesítésének eredménye (3). Végül mindenképpen szükséges a (6) és (9) klánokat is egyesíteni.

#### MIÉRT INFORMATIKA?

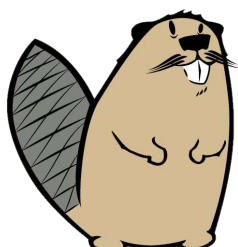
Ennek a feladatnak a megoldása egy optimalizálási probléma - bizonyos értéket (ebben az esetben az egyesülés időtartama) szeretnénk minimalizálni vagy maximalizálni. Ehhez általában jól megírt algoritmusokat használunk (hiszen nem csak ilyen kevés, adattal dolgozhatunk). Sokszor nem is találjuk meg megadott idő alatt az optimális megoldást és megelégszünk egy közel megfelelővel.

Ilyen probléma a mindennapi életünkben például az erőforrás-takarékos, környezetbarát termelési folyamatok menedzselése is.

Ebben az esetben még egy egyszerű mohó algoritmus is megoldhatja a problémát. Ekkor megoldhatjuk a feladatot lépésről lépésre egy többnyire egyszerű ötlettel (ebben az esetben "először egyesítsük a kis klánokat") anélkül, hogy valaha is módosítanunk kellene egy döntést.

#### WEBOLDAL

[https://hu.wikipedia.org/wiki/Moh%C3%B3\\_algoritmus](https://hu.wikipedia.org/wiki/Moh%C3%B3_algoritmus)



## HÍDÉPÍTÉS (2021-PK-07)

## KISHÓD - KÖNNYŰ

Gabi szeretne egy hidat építeni.

Ehhez szüksége van kalapácsra, szögre, deszkára és kötéltre.

A pincében talált is kalapácsot és köteleket. A többi szükséges holmit pedig meg kell vásárolnia.

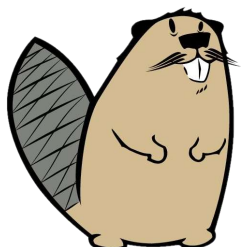


Három üzlet kínálatát látod a képeken.



## Hova menjen el Gabi vásárolni?

- A) Balról az első és a második üzletbe.
- B) Balról a második és a harmadik üzletbe.
- C) Balról az első és a harmadik üzletbe.
- D) Mindhárom üzletbe.



A helyes válasz az A)



A bal oldali üzletben a hídépítéshez két szükséges holmit is árulnak: a kalapácsot és a szögeket. Gabinak már megvan a kalapácsa, de itt megvásárolhatja a szögeket (amire szüksége van és semelyik másik boltban nem találhatóak meg).

A másik hiányzó holmi a deszka, melyet csak a második üzletben vehet meg, így oda is be kell mennie. Több hiányzó holmi nincs a listán, így a harmadik üzletbe nem kell bemennie.

#### MIÉRT INFORMATIKA?

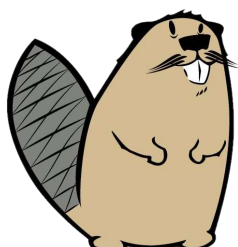
Ebben a hódfeladatban lévő üzletek összesen hét dolgot árulnak: ollót, kalapácsot, szöget, téglát, deszkát, kötelet és festéket. Ez elég sok! A két dolog, amit Gabinak meg kell vásárolnia, része ennek a felsorolásnak. Egy általánosított megoldás valahogy így is leírható: Vesszük a teljes készlet összes elemét és minden egyes tételnél megjelöljük, hogy Gabi beszerzési részalmazához tartozik -e vagy sem.

Ezután minden üzletre külön-külön bejelöljük, hogy árulják-e az adott terméket vagy sem. Már csak páronként össze kell hasonlítaniunk a beszerzési listát az egyes üzletekben árult termékekkel.

A számítógépes programoknak is gyakran össze kell hasonlítaniuk a mennyiségeket. Ezt a következőképpen tehetik meg: Egy bit szükséges minden dologhoz, ami előfordulhat a listában. A számítógép két érték egyikét tárolhatja a bitben, például az „igen” és a „nem”. Ebben az esetben mentésre kerül, hogy a dolog egy halmazhoz tartozik-e („igen”) vagy sem („nem”). Ezután egy program összehasonlíthat két halmazt: ellenőrzi, hogy az egyik halmaz bitje „igen”, és a másik halmaz azonos bitje is „igen”. Egy számítógép különösen gyorsan képes ilyen ellenőrzések elvégzésére. A bitekkel rendelkező halmazok leírása ezért nagyon gyakori az informatikában.

#### WEBOLDAL

[http://tehetseg.inf.elte.hu/szakkorefop2017/pdf/ELTEIKSzakk%C3%B6r\\_Halmaz%20t%C3%ADpus\\_20181101.pdf](http://tehetseg.inf.elte.hu/szakkorefop2017/pdf/ELTEIKSzakk%C3%B6r_Halmaz%20t%C3%ADpus_20181101.pdf)

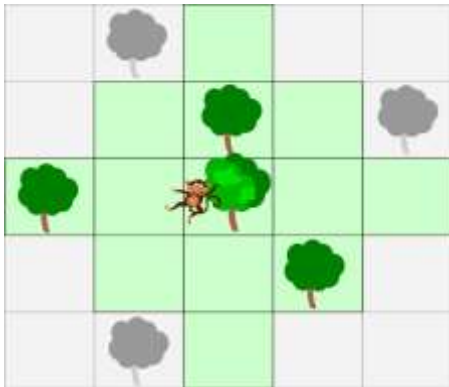


## KOKO (2021-SI-02)

JUNIOR – NEHÉZ

SENIOR – KÖZEPES

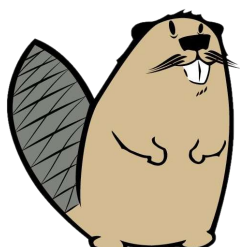
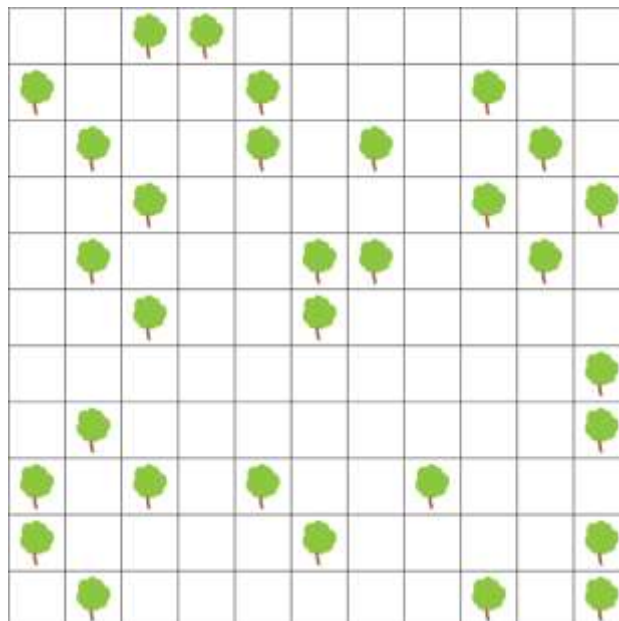
Koko, a majom egy fáról egy ugrással a képen a zöld négyzettel jelölt helyekre juthat el.



A zöld fákra is eljuthat egy ugrással. Ezekről (konkrétan a legfelső zöld fáról) egy második ugrással eljuthat a felső két szürke fára is.

Vannak olyan facsoportok, amelyek között Koko tetszés szerint több ugrással is mozoghat anélkül, hogy akár csak egyszer is érintenie kellene a talajt.

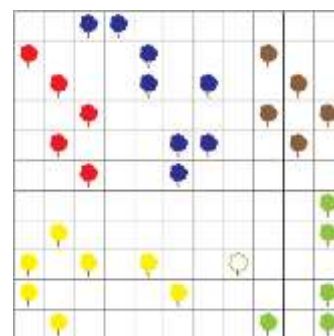
**Hány fából áll a legnagyobb ilyen facsoport a képen:**



## A helyes válasz: 8 fából

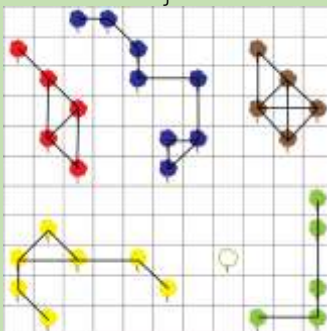
A képen azonos színekkel jelöltük az egy facsoportba tartozó fákat, melyeken Koko tetszés szerint több ugrással is mozoghat anélkül, hogy érintenie kellene a talajt.

Megszámolva beláthatjuk, hogy a legnagyobb facsoport, a kék, 8 fából áll.



### MIÉRT INFORMATIKA?

Ha Koko át tud ugrani az egyik fáról a másikra, akkor gyakorlatilag ezek a fák össze vannak kötve egymással. Ezt a fák közötti vonalként ábrázolhatjuk:



Tehát van egy gráfunk, ahol a fák a csomópontok és élek jelölik az "átugorható" kapcsolatot ezek között. Koko átjuthat egy vagy több ugrással egyik fáról a másikra, ha az élek utat képeznek a két fa között.

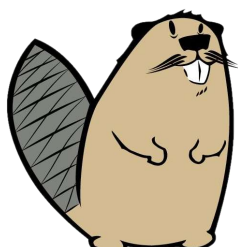
A csomópontok egy csoportját összefüggőnek nevezzük, ha mindegyiket élek kötik össze a többivel. Ha egy ilyen csoportot már nem tudunk további csomóponttal bővíteni anélkül, hogy elveszítenénk az összefüggőséget, akkor "összefüggő komponensről" beszélünk. Egy gráf egyértelműen felosztható az ilyen összefüggő komponensekre, ezeket jelöltük az ábrán különböző színekkel.

Az összefüggő komponens könnyen meghatározható úgy, hogy bármely csomópontból kiindulunk, majd megkeressük az összes csomópontot, amely az éleken keresztül elérhető.

Ez a problémakör kapcsolódik a hálózati folyam-problémákhoz: egy gráf összefüggősége, a hálózat hibákkal szembeni ellenálló képességének fontos mértéke.

### WEBOLDAL

[https://hu.wikipedia.org/wiki/%C3%96sszef%C3%BCgg%C5%91s%C3%A9g\\_\(gr%C3%A1fm%C3%A9let\)](https://hu.wikipedia.org/wiki/%C3%96sszef%C3%BCgg%C5%91s%C3%A9g_(gr%C3%A1fm%C3%A9let))





## KULCSTARTÓK (2021-SK-01)

KISHÓD - KÖZEPES

BENJAMIN - KÖNNYŰ

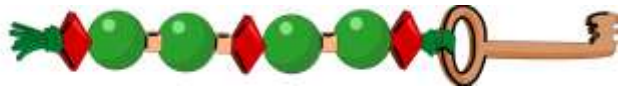
ANNA, BELLA és LENA a nevüket rejtő kulcstartót készítene. A nevük betűihez kétféle gyöngyöt

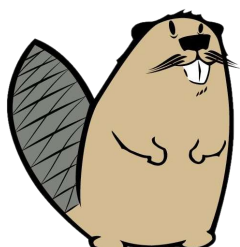
használnak:  és . Az egyes betűket pedig a  gyönggyel választják el.

Két kulcstartó már el is készült:

|                    |                                                                                    |
|--------------------|------------------------------------------------------------------------------------|
| Anna kulcstartója  |  |
| Bella kulcstartója |  |

**Vajon melyiket készítette Lena**

|   |                                                                                      |
|---|--------------------------------------------------------------------------------------|
| A |  |
| B |  |
| C |  |
| D |  |





## A helyes válasz az A)

A LENA szó L betűvel kezdődik. A BELLA -ban az L a harmadik és a negyedik betű, és a ezeket a



gyöngyök jelentik.

Csak az A) és a D) válasz kezdődik ezzel a betűvel. A LENA második betűje a BELLA második betűje,

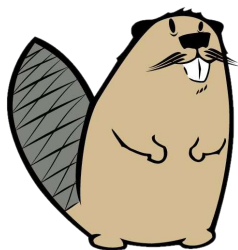
az E, ami a  gyöngy. Mind az A), mind a D) esetében ez a második betű, így mindkettő továbbra is megoldás lehet.

Ezután találjuk ki az N betű gyöngyeit. ANNA kulcstartóján megtaláljuk az N gyöngyeit:  . És ezek a következő gyöngyök az A) megoldásban.

Egy másik lehetőség, hogy meghatározzuk a gyöngyöket a LENA kulcstartójához, ha táblázatot készítünk a gyöngyökkel a már ismert betűkhöz. ANNA kulcstartójából az A és az N gyöngyeit, BELLA kulcstartójából pedig a B, az E és az L betűkhöz tartozó gyöngyöket határozhatjuk meg. Ezután már kialakíthatjuk LENA kulcstartóját, vagy akár a többi megoldást is: BENA (B kulcstartó), NENA (C kulcstartó) és a LEAN (D kulcstartó) kifejezéseket.

Kódolási táblázat:

| Betűk | Gyöngyök                                                                                                                                                                                                                                                                                                                                        |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| A     |                                                                                                                                                                           |
| N     |                                                                                                                                                                           |
| B     |     |
| E     |                                                                                                                                                                                                                                                              |
| L     |     |



## MIÉRT INFORMATIKA?

Az információt már nagyon régóta több ok miatt is igyekszünk kódolni:

- azért, hogy bizonyos feltételek mellett üzeneteket tudjunk továbbítani, vagy
- titokban (titkosítva) tartsuk az üzeneteink értelmét.

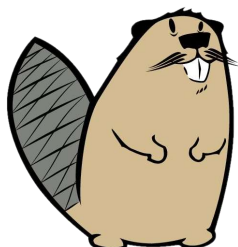
Ebben a hód feladatban a kódolás Morse-kódon alapul. A Morse-kód pontját (•) a kerek , a kötőjelet (-) pedig a  gyöngy jelképezi. Az A betű Morse-kódban: • –

Bármilyen szöveg kódolásához szükségünk van az ábécé minden betűjének kódjára. A Morse-kódot a XIX. században kezdték el használni. 1837-ben Samuel Morse feltalált egy egyszerű elektromágneses távirót. Az akkor használt kód csak a tíz számjegyből állt; az átadott számokat táblázat segítségével betűkké és szavakká kellett fordítani. Alfred Lewis Vail, Morse munkatársa 1838-ban dolgozta ki az első olyan kódot, amely betűket is tartalmazott. A kódot szövegek akusztikus, vizuális vagy elektromos továbbítására fejlesztették ki távíróvonalakon keresztül. Egy pont rövid átviteli idő, egy vonal pedig háromszor hosszabb. A betűk között szünetet kell tartani. Hosszú szünet választja el a szavakat. A morze kódot ma is használják például SOS vészjelzésként. Az SOS Morse-kódban (3x rövid, 3x hosszú, 3x rövid) nagyon könnyen továbbítható kiabálással, kopogással vagy a zseblámpa fényével. Az elektronikus adatfeldolgozás során a karaktereket általában számértékkel kódolják, hogy továbbítsák vagy tárolják őket.

## WEBOLDALAK

<https://hu.wikipedia.org/wiki/Morzek%C3%B3d>

<https://hu.wikipedia.org/wiki/Karakterk%C3%B3dol%C3%A1s>



## ISKOLAI ASZTALOK (2021-SV-01)

KADÉT - NEHÉZ

JUNIOR – KÖZEPES

SENIOR – KÖNNYŰ

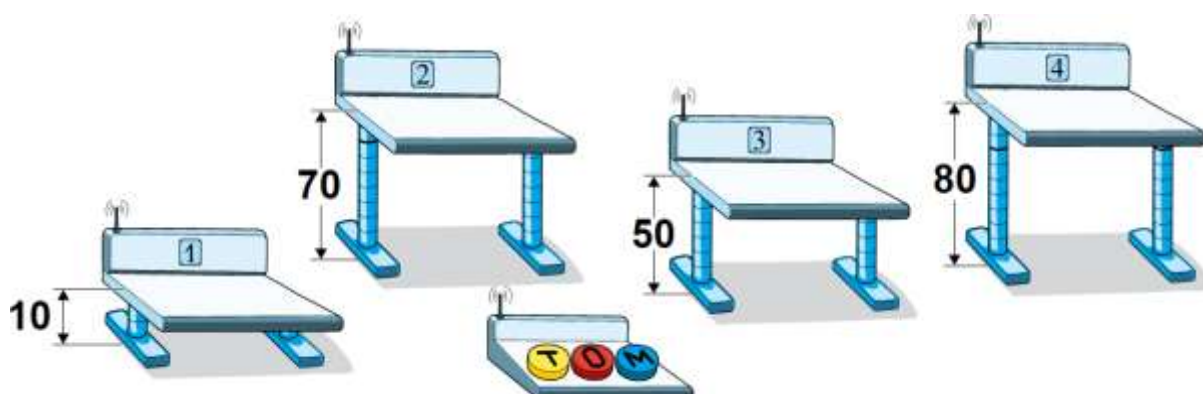
Az osztályteremben állítható magasságú asztalok találhatók. A tanításhoz minden asztalt 60 cm magasságra kell állítani. Ehhez egy háromgombos távirányítót használhatunk.

Valaki játszott a távirányítóval és átprogramozta. Most a három gomb a következőképpen működik:

-  az 1, 2 és 3 asztalok magasságát egyenként 10 cm-rel növeli.
-  a 2, 3 és 4 asztalok magasságát 10 cm-rel lejjebb engedi.
-  az 1, 3 és 4 asztal magasságát 10 cm-rel növeli.

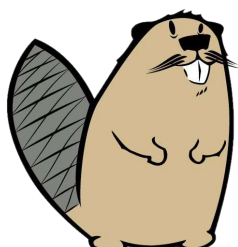
Az egyes gombokhoz rendelt feladatok mindegyikét minden esetben végre kell hajtani, amikor azt a gombot megnyomjuk.

Jelenleg az 1, 2, 3 és 4 asztalok magassága 10 cm, 70 cm, 50 cm és 80 cm:



### Hogyan tudjuk beállítani mind a négy asztal magasságát 60 cm-re?

- A) Nyomjuk meg 4-szer , 5-ször  és 1-szer .
- B) Nyomjuk meg 5-ször , 1-szer  és egyszer sem .
- C) Nyomjuk meg 3-szor , 4-szer  és 2-szer .
- D) Nyomjuk meg 2-szer , 4-szer  és 6-szor .



## A helyes válasz a C)

Nyomjuk meg 3-szor , 4-szer  és 2-szer .

A távirányító mindhárom gombjának megnyomásakor a magasság 10 cm-rel változik, azaz mindig ugyanannyival. A gombok közül kettő felemeli az asztalokat ( és ) és csak egy gomb engedi le azokat (). Ezenkívül mindhárom gomb három-három asztal magasságát változtatja, így mindig van egy asztal, amelynek magassága változatlan marad.

Az 1. asztal 50 cm-rel alacsonyabb, mint szükséges. Ebből következik, hogy a  vagy a  gombot pontosan ötször kell megnyomnunk (összesen és együttesen 5-nek kell lennie). Ezt egyenletként leírhatjuk így is:  $T + M = 5$ .

A  gomb nincs hatással az 1. asztalra, így az 1. asztalt egyáltalán nem tudjuk lefelé mozgatni. Ezzel kizárhatjuk a D) megoldást, mert ott  $T + M = 8$ . Hiszen ekkor az 1. asztal magassága  $10 + 20 + 60 = 90$  cm (kezdő magasság +  $2 \times 10 + 6 \times 10$  esetén).

A 2. asztal 10 cm-rel magasabb, mint kellene.  nincs hatással a 2. asztalra. Így  $T - O = -1$ . Ezzel kizárhatjuk a B) megoldást (a 2. íróasztal magassága ekkor  $70 + 50 - 10 = 110$ ) lenne.

A 3. asztal 10 cm-rel alacsonyabb, mint kellene, ezért a következők érvényesek:  $T - O + M = 1$ . Ez azt jelenti, hogy az A) és B) megoldás kizárható. A) a 3. asztal magassága változatlan ( $50 + 40 - 50 + 10 = 50$ ); B) az íróasztal magassága 3:  $50 + 50 - 10 = 90$ .

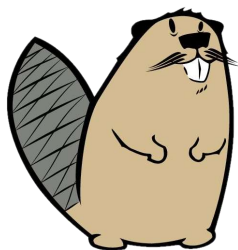
Most már minden megoldás kizártunk, kivéve a C)-t.

Azonban még ellenőrizni kell, hogy a C) megoldás megfelelő magasságot ad-e a 4-es asztalnak. A 4-es asztal 20 cm-rel magasabb,  gombnak nincs hatással a 4-es asztal magasságára. Tehát  -t kétszer kell megnyomni, és a  gomb minden egyes megnyomására újra meg kell nyomni  -t. A C) megoldás nyomogatásait követve a 4. asztal magassága:  $80 - 40 + 20 = 60$ .

Mivel már megállapítottuk, hogy a C) megoldás az 1., 2. és 3. asztalon működik, biztosak vagyunk benne, hogy ez a megoldás működni fog.

Alternatív módon a megoldás megtalálható 4 lineáris egyenlet használatával. Minden asztalhoz ír le egy egyenletet, mely billentyűk változtatják az asztal magasságát, és milyen magasságváltozást keresel. Például az 1. asztal magassága csak a  és a  gombbal változik, a kívánt magasságállítás 50 cm, ami 5 gombnyomással érhető el (mert 10 cm gombnyomásonként).

Mivel négy asztal és három gomb található, négy lineáris egyenlet van három ismeretlennel:



$$\left. \begin{array}{rcl} T + M & = & 5 \\ T - O & = & -1 \\ T - O + M & = & 1 \\ -O + M & = & -2 \end{array} \right\}$$

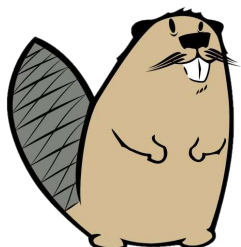
Ha kivonjuk a harmadik egyenletet az elsőből, akkor  $O = 4$  értéket kap. A második egyenletbe illesztve  $T = 3$  -at kapunk. Csak  $M = 2$  esetén teljesül minden egyenlet. Tehát ez az egyetlen megoldás.

#### MIÉRT INFORMATIKA?

Ez tipikus feladat a diszkrét optimalizálás, pontosabban a lineáris programozás területén. Számos korlátozás kerül megadásra, és ebben a speciális esetben mindegyik lineáris egyenletként is megfogalmazható. A cél, amitől ez informatikai feladattá válik, hogy az ember egy adott célhoz vezető cselekvéssorozatot keres.

Az egész feladatot emiatt úgy is leírhatnánk, mint egy út keresését a 4 dimenziós térben, három engedélyezett mozgási akcióval, nevezetesen a  $(10,70,50,80)$  ponttól a  $(60,60,60,60)$  pontig.

Erre a feladatra csak egy megoldás létezik, de az ilyen feladatoknak gyakran sok megoldása van, ami lehetővé teszi az optimalizálást, mint célkitűzést. Ezután keressük a  $T + M + O$  lineáris függvény minimumát.



## MOLNÁR MONA (2021-TR-06)

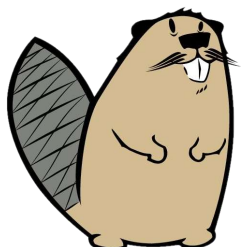
KISHÓD - KÖZEPES  
BENJAMIN - KÖNNYŰ

Molnár Monának három malma van.

Jelenleg azonban sok víz folyik el feleslegesen. Molnár Mona szeretné, ha a víz csak a saját malmait hajtáná, egy cseppet sem szeretne elpazarolni belőle. Az ábrán látható gátakkal tudja szabályozni, hogy a víz melyik mederben folyjon a nyíl irányába.



**Legkevesebb hány gátat zárjon, hogy csak a malmait hajtsa meg a víz?**

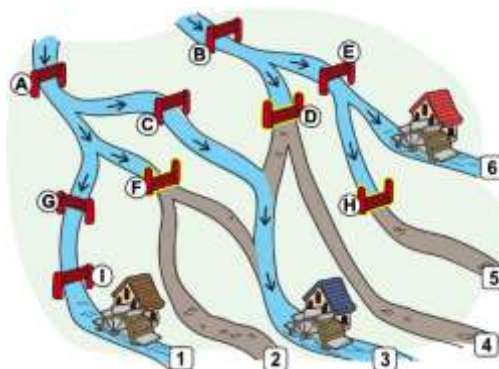


## A helyes válasz: legkevesebb 3 gátat

Ezek a következő rajzon D, F és H jelzéssel szerepelnek.

Csak így lehet vízzel ellátni a három malmot további vízpazarlás nélkül:

- Az A, G és I gátaknak nyitva kell lenniük, különben a víz nem folyik a malomhoz (1-essel jelölt meder).
- A B és E gátakat is nyitva kell tartani, különben a 6-os medernél található malom nem kap vizet.
- Mivel az A gát nyitva van, az F gátat zárni kell, különben a 2-es mederben a víz kárba vész, hiszen itt nincs malom.
- Hasonlóképpen, mivel a B gát nyitva van, a D gátat zárni kell, különben a 4-es mederbe kerülő víz kárba vész.
- A C gátak miatt nyitva kell lennie, különben a 3-as mederben a malom nem jut vízhez.
- Végül a H gátak zárva kell maradnia (mivel a B és E gát nyitva van), különben a vizet az 5-ös medernél elpazaroljuk.



### MIÉRT INFORMATIKA?

Ebben a feladatban a víz áramlását a feltételek szabályozzák. Például a víz akkor folyik a 6-os mederben található malomhoz, amikor a B és az E gát nyitva van. A víz akkor folyik a 3-as mederben található malomhoz, ha a következő feltételek közül az egyik (vagy mindkettő) teljesül:

1. az A gát nyitva van, és a C vagy az F gát közül az egyik nyitva van, vagy
2. a B és a D gát nyitva van.

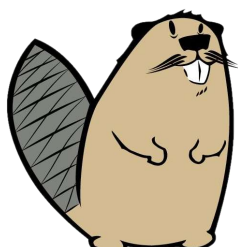
Ezeket a kombinált feltételeket az "és" (AND,  $\wedge$ ) vagy a "vagy" (OR,  $\vee$ ) logikai operátorok használatával valósíthatjuk meg, állíthatjuk össze. Az ilyen operátorok olyan igazságértékeket kombinálnak, mint az igaz vagy a hamis. Tehát ha A és B két igazságérték, akkor tudjuk, hogy az „A ÉS B” vagy „A VAGY B” összetett kifejezések mely igazságértékekkel rendelkeznek:

Az informatikában (és a matematikában is) az „A VAGY B” állítást akkor tekintjük igaznak, ha vagy A, vagy B igaz. Az „A ÉS B” állítást pedig akkor tekintjük igaznak, ha mind A, és mind B igaz.

Programozáskor fontos a feltételek helyes megfogalmazása. A feltételes utasítások (elágazások), valamint a feltételes ismétlések esetén (ciklus közben) a programfolyamat szabályozására feltételek szolgálnak. A logikai operátorokkal való kapcsolatok hasznosak összetettebb feltételek megfogalmazásában.

### WEBOLDALAK

[https://hu.wikipedia.org/wiki/Felt%C3%A9teles\\_utas%C3%ADt%C3%A1s](https://hu.wikipedia.org/wiki/Felt%C3%A9teles_utas%C3%ADt%C3%A1s)  
[https://hu.wikipedia.org/wiki/Ciklus\\_\(programoz%C3%A1s\)#Felt%C3%A9teles\\_ciklus\\_\(while\)](https://hu.wikipedia.org/wiki/Ciklus_(programoz%C3%A1s)#Felt%C3%A9teles_ciklus_(while))  
[https://hu.wikipedia.org/wiki/Logikai\\_m%C5%B1velet](https://hu.wikipedia.org/wiki/Logikai_m%C5%B1velet)



## A LEGHOSSZABB SOROZAT (2021-UA-01B)

KISHÓD - NEHÉZ

BENJAMIN - KÖZEPES

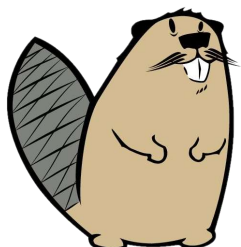
KADÉT - KÖNNYŰ

Adott egy 16 elemű és kétféle alakzatból felépülő sorozat, ahogy az a képen is látható:



Pontosan két alakzatot változtathatsz meg!

**Mi a lehető leghosszabb, csak azonos alakzatból álló részsorozat hossza?**





## A helyes válasz a 8

Ahhoz, hogy ezt megmutassuk, két dolgot kell bizonyítanunk: (1) hogy van egy 8 hosszú, azonos alakzatokból álló részsorozat, és (2) hogy nem létezik 8-nál hosszabb ilyen részsorozat.

Az első részt könnyű bizonyítani. Például így lehet egy 8 csillagból álló részsorozatot készíteni:



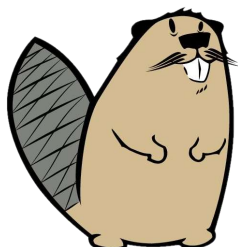
Annak bizonyítására, hogy egy 8-nál hosszabb azonos alakzatból álló sorozat létrehozása nem lehetséges, fontoljuk meg hogy létrehozható-e egy 9 hosszú. Mivel csak két alakzatot változtathatunk meg, ezért a 9 hosszúságú részsorozatnak, már az eredeti sorozatban is, hét azonos alakzattal kell rendelkeznie.

Nyolc darab 9 hosszú részsorozat van az eredeti sorozatban, ezek közül néhányat az alábbiakban mutatunk be. *Egyetlen részsorozatban sem található hét azonos alakzat.* Nyugodtan győződj meg erről magad is a fennmaradó négy, alul nem látható részsorozat esetében is.



Mivel nem lehet megszakítás nélküli 9 hosszú azonos alakzatokból álló részsorozatot létrehozni, ezért természetesen nem lehetséges, hogy az azonos alakzatokból álló részsorozat hossza nagyobb legyen, mint 9.

Így megmutattuk, hogy a lehető leghosszabb, azonos alakzatokból álló részsorozat hossza 8.



## MIÉRT INFORMATIKA?

Ez a hódfeladat egy bizonyos feltételeknek megfelelő *leghosszabb karakterlánc* megtalálása.

Az informatikában sok olyan eset van, amikor a leghosszabb karakterlánc megtalálása hasznos, különösen a leghosszabb közös karakterlánc (szöveg) megtalálása két karakterlánc (szöveg) esetében.

Segíthet a plágium észlelésében és az adatok duplikációval (az adatok redundáns másolatainak eltávolítása) történő tömörítésében vagy mentésében. Például egy digitális archívumban a duplumok, vagyis a többszörösen lementett azonos tartalmak eltávolításában vagy minimalizálásában. Ezzel a tárhelyigény és a felhasználók többszörös találatai ugyanarra a tartalomra is csökkenthetők.

A leghosszabb szekvenciák megtalálásához használható technikák közé tartozik a két mutató módszer és a csúszó ablak algoritmus.

## WEBOLDALAK

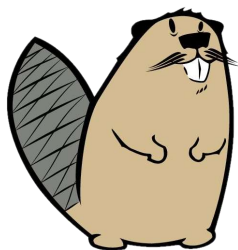
<https://hu.wikipedia.org/wiki/Pl%C3%A1gium>

<https://hu.wikipedia.org/wiki/String>

[https://hu.wikipedia.org/wiki/Mutat%C3%B3\\_\(programoz%C3%A1s\)](https://hu.wikipedia.org/wiki/Mutat%C3%B3_(programoz%C3%A1s))

<https://hun.fitforlearning.org/860759-what-is-sliding-window-algorithm-YKWXDU>

<https://gyires.inf.unideb.hu/GyBITT/30/ch03s04.html>



## DÖŐŐŐL A FA! (2021-UY-11)

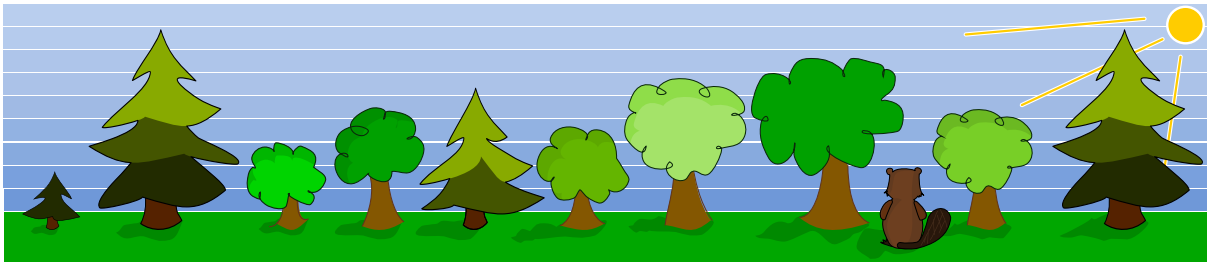
KISHÓD - NEHÉZ

BENJAMIN - KÖZEPES

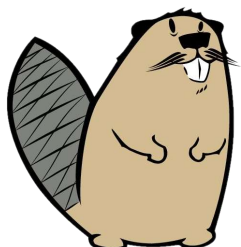
KADÉT - KÖNNYŰ

Egy hód gátat akar építeni. Azért, hogy mindig a megfelelő fákat vágja ki, két feltételt fogalmazott meg és csak akkor vág ki egy fát, ha mindkét feltétel teljesül:

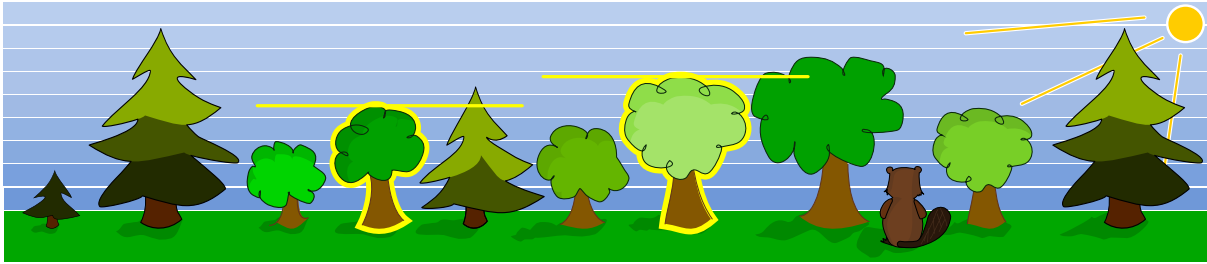
- A kivágandó fa mellett közvetlenül balra van egy alacsonyabb fa.
- A kivágandó fa mellett közvetlenül jobbra van egy magasabb fa.



**Hány fát vág ki a képen látható fasorból?**



A helyes válasz: 2 fát



Csak balról (és jobbról is 😊) a negyedik és hetedik fa felel meg mindkét megadott feltételnek egyszerre: közvetlenül a bal oldalán van egy alacsonyabb ÉS közvetlenül a jobb oldalán egy magasabb fa.

#### MIÉRT INFORMATIKA?

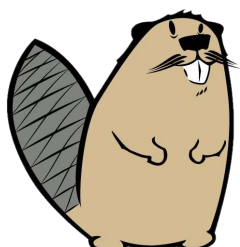
Az informatika gyakran olyan problémák megoldásáról szól, amelyeket egy sor logikai megkötés határoz meg. A feladat az, hogy olyan megoldást találjunk, amely megfelel az összes megkötésnek.

Ennél bonyolultabb feladatok is megoldhatóak ha a megszorításokat logikai operátorok segítségével kombináljuk. A "konjunkció" ( $\wedge$  vagy ÉS operátor) például az  $A \wedge B$  kifejezésben akkor és csak akkor ad vissza igaz értéket, ha mindkét korlátozás igaz.

Ez az alapelv az informatika szinte minden területén megtalálható, a programozástól az adatok, adatbázisok kezeléséig.

#### WEBOLDAL

[https://hu.wikipedia.org/wiki/Logikai\\_m%C5%B1velet](https://hu.wikipedia.org/wiki/Logikai_m%C5%B1velet)



## MENTS MEG A FÁT! (2021-UZ-02)

BENJAMIN - NEHÉZ

KADÉT - KÖZEPES

JUNIOR – KÖNNYŰ

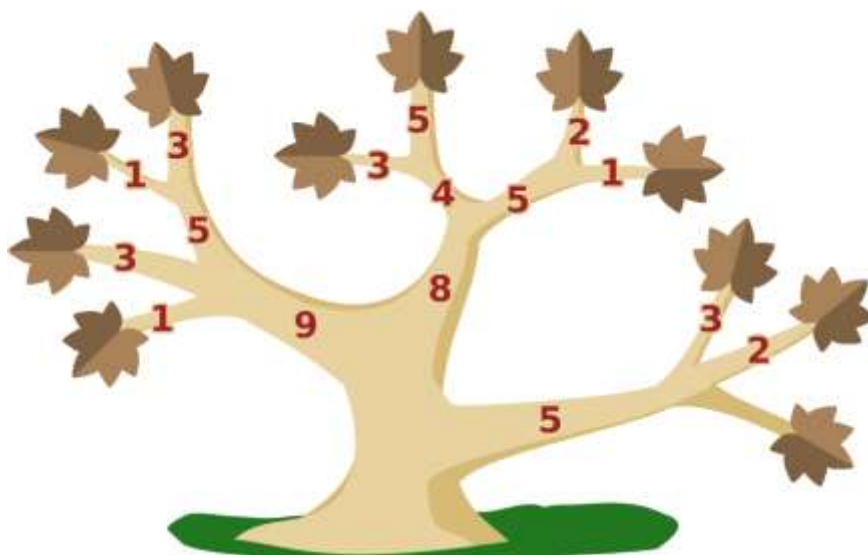
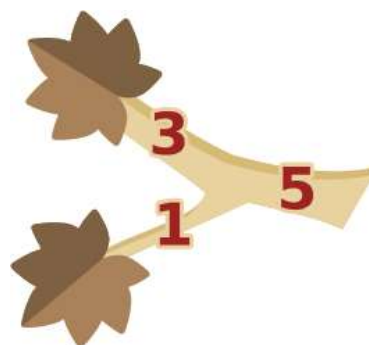
Bruno kertjében egy fa beteg, minden levele kiszáradt. Bruno meg akarja menteni a fát. Ehhez le kell fűrészelnie néhány ágat, hogy a végén ne maradjon levél a fán. Ezután új ágak nőhetnek új levelekkel az így meggyógyított fán.

A képen egy példa látható:

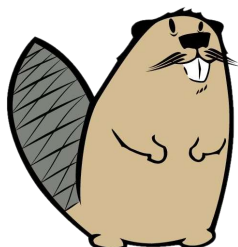
A két levél eltávolításához Bruno vagy lefűrészezi a két ágat a levelekkel, vagy csak azt az egy ágat, amelyről a másik kettő elágazik.

Minden ág esetében a számok jelzik, hogy mennyi ideig tart a fűrészelés. Bruno mielőbb készen akar lenni, tehát lefűrészezi a két ágat a levelekkel, mivel  $3 + 1 < 5$ .

Itt látható a teljes fa.

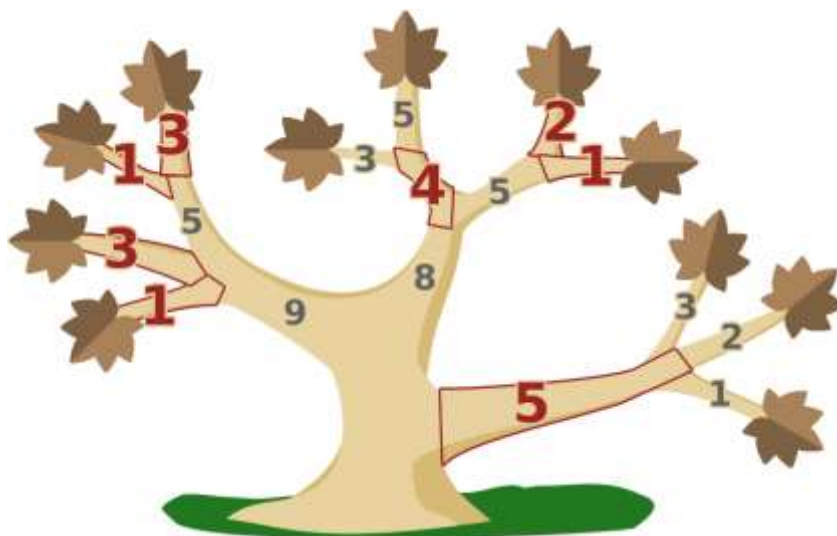


**Legfeljebb mennyi ideig tart a fűrészelés, ha Brúnó a lehető leggyorsabban szeretne végezni?**



## A helyes válasz 20 hódóra

Bruno ahhoz, hogy a leggyorsabban készen legyen, a pirossal megjelölt ágakat fűrészelni le.



De miért van ez így? Először is kiszámíthatjuk, mennyi időre lenne szüksége Brunónak, ha csak levágja a közvetlen leveles ágakat:

$$1 + 3 + 1 + 3 + 3 + 5 + 2 + 1 + 3 + 2 + 1 = 25$$

Most haladjunk a törzs irányába, és újra és újra gondolkodjunk el azon, hogy gyorsabb lenne-e lefűrészelni azt az ágot, amelyről az előző ágak közvetlenül vagy közvetve elágaznak.

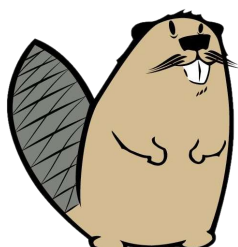
Az első ilyen lépés után a következő számítási eredmények (ahol a "min" függvény kiszámítja az egyes fűrészelési idők minimumát):

$$1 + 3 + \min(5, 1 + 3) + \min(4, 3 + 5) + \min(5, 2 + 1) + \min(5, 3 + 2 + 1) = 1 + 3 + 1 + 3 + 4 + 2 + 1 + 5 = 20$$

De ekkor még nem végeztünk, hiszen vannak még lehetséges összevonások, azaz ahol a fűrészelési idők minimumát kell egy újabb elágazás miatt vizsgálnunk. Ez után a lépés után már megérkeztünk a törzshöz:

$$\min(9, 1 + 3 + 1 + 3) + \min(8, 4 + 2 + 1) + 5 = 1 + 3 + 1 + 3 + 4 + 2 + 1 + 5 = 20$$

Brunó nem tudja gyorsabban befejezni a munkát.



## MIÉRT INFORMATIKA?

Képzeld el, hogy Bruno fájának lefűrészelt darabjai nem estek közvetlenül a földre. Akkor azt mondhatnánk, hogy a fát csak két részre vágják a fűrészeléssel: Az egyik rész tartalmazza az összes lefűrészelt darabot és természetesen a leveleket, a másik részben marad a törzs és az összes ág, amely addig a fűrészelési pontig terjed. Ez a szétbontás vagyis ez a „favágás” minimális azt az időt nézve, amelyet Bruno a fűrészeléssel tölt.

Az informatika is ismeri a fákat, és azok segítségével ábrázolja azokat az objektumokat, amelyek bizonyos módon kapcsolódnak egymáshoz. Az objektumokat csomópontoknak, a kapcsolatokat éleknek nevezzük. Mindig csak egy út van két csomópont között az élek mentén - csakúgy, mint egy valódi fában egy levélből vagy egy ág elágazásából, mindig csak egy út van az ágak mentén a törzsig. Ha e feltételtől eltekintünk, általánosabban egy gráfról beszélünk.

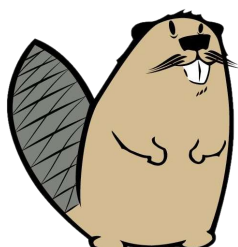
Egy általános gráfban a minimális vágást, azaz a két vagy több részre bontást minimális költségekkel nem olyan könnyű kiszámítani, mint amit itt egy fánál mutattunk, de nem is túl nehéz. Ez jó, mert érdekes felhasználási lehetőségei vannak. Minimális vágások használhatók például a képfájlok hasonló részekre történő felosztásakor. Speciális gráfok az áramlási hálózatok, amelyekkel többek között a hálózatokon keresztül történő adatáramlások is modellezhetők, a minimális vágás költségei a teljes hálózaton keresztül lehetséges maximális áramlásnak felelnek meg.

## WEBOLDALAK

[https://hu.wikipedia.org/wiki/Fa\\_\(gr%C3%A1fm%C3%A9let\)](https://hu.wikipedia.org/wiki/Fa_(gr%C3%A1fm%C3%A9let))

[https://hu.wikipedia.org/wiki/Maxim%C3%A1lis\\_%C3%A1raml%C3%A1si\\_probl%C3%A9ma](https://hu.wikipedia.org/wiki/Maxim%C3%A1lis_%C3%A1raml%C3%A1si_probl%C3%A9ma)

[https://hu.wikipedia.org/wiki/Maxim%C3%A1lis\\_folyam\\_%E2%80%93\\_minim%C3%A1lis\\_v%C3%A1g%C3%A1s](https://hu.wikipedia.org/wiki/Maxim%C3%A1lis_folyam_%E2%80%93_minim%C3%A1lis_v%C3%A1g%C3%A1s)

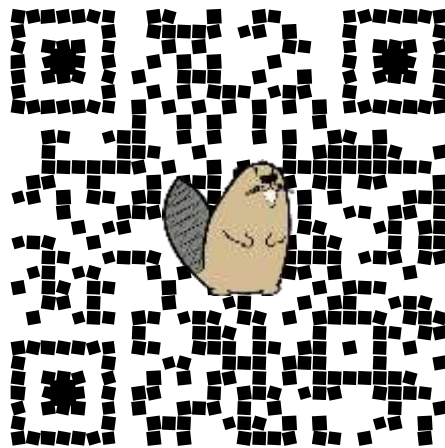


## TÁMOGATÓINK, KÖSZÖNETNYÍLVÁNÍTÁS

Köszönjük a nemzetközi Bebras kezdeményezés országainak, kiemelten a DACH-csapatnak  
(Németország, Ausztria, Svájc),  
az ELTE IK hallgatóinak, illetve  
a kapcsolattartó tanároknak szervezői munkáját.

A HÓD VERSENY MINDEN TARTALMÁRA A CC BY-NC-SA 4.0 LICENSZ VONATKOZIK.

*A verseny kérdései Kutatási célokra csak a szervezők megkérdezésével, megkeresésével és  
kutatási munkájuk ismeretével, valamint idézésével használhatóak fel.*



ELTE | IK  
INFORMATIKAI KAR

