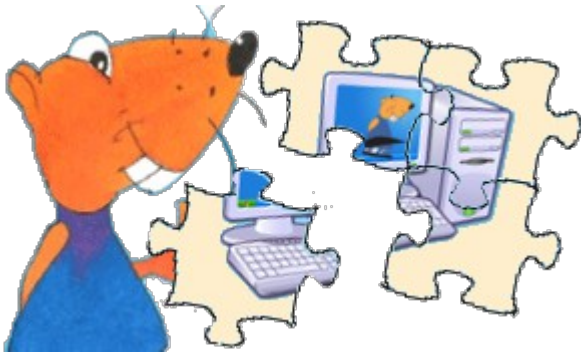




**Bebras – Beaver – Beber – Bever – Bobr – Bobor –
Bobřík – Kobras – Majava – Castoro – Castor –
Бобер – Hód**

**Bebras: Nemzetközi informatikai és számítógép-
késztség verseny (International Contest on
Informatics and Computer Fluency) –
MINDENKINEK**

Az *e-HÓD/HÓD*ítsd meg a biteket a BEBRAS-kezdemenyezés magyar partnere.

A Bebras Dr. Valentina Dagiene litván professzor által életre keltett verseny, mely a nemzetközi Kenguruhoz hasonló célokkal rendelkezik, de nem a matematika, hanem az informatika területén. Bebras litvánul hódót jelent.

A verseny 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 félelmeket, negatív érzéseket;
- megmutassa az informatika területének 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.

A versenyt négy korcsoport számára rendezik:

- 5. és 6. osztály, Benjamin
- 7. és 8. osztály, Meteor
- 9. és 10. osztály, Junior
- 11. és 12. osztály, Senior.

Magyarországon 2014-ben negyedik alkalommal, mind a négy korcsoportban meghirdettük a megmérettetést.

A versenyt az ELTE IK T@T Labor és az NJSZT Közoktatási Szakosztálya szervezi.

Az alábbi dokumentumban a 2014-es magyar verseny feladatai és megoldásai találhatóak.

További információkért látogasson el a <http://e-hod.elte.hu/> weboldalra, vagy írjon e-mail-t az info@e-hod.elte.hu címre.

Résztétel

A résztétel mindenki számára ingyenes.

A verseny november második hetében kerül lebonyolításra, osztályonként kiválasztható, hogy az adott héten melyik napon mikor (reggel 8:00-tól délután 2-ig). Ezzel biztosítható, hogy akár egy-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 átbeszélhetőek ugyancsak akár egy tanóra keretein belül.

Szabályok

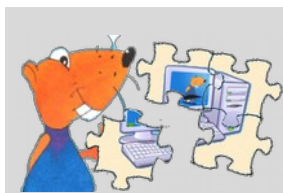
- a résztvevők online kapják meg és válaszolják meg a kérdéseket;
- a versenyre fordítandó idő 45 perc, 18 feladat három nehézségi szinten: könnyű, közepes és nehéz;
- a verseny alatt semmilyen más számítógépes program, alkalmazás nem használható;
- a verseny során nyugalmas környezetet kell biztosítani;
- 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 megválaszolhatóak;
- a kérdések, problémák megértése a feladat részét képezi. Ezért a feladatok megbeszélése, értelmezéssel kapcsolatos kérdések nem megengedettek;
- a verseny befejezése után, a hód hetet követően kerülnek nyilvánosságra a megoldások;

Értékelés, pontozás

Minden korcsoportban 18 feladatot kell megoldani három nehézségi szinten. Minden helyes válasz pontot ér, minden helytelen válaszért 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

Minden résztvevő kezdetben 54 pontot kap. Így összesen maximum 216 pontot érhet el, illetve 0-ra csökkentheti pontjait, amennyiben minden kérdésre helytelen választ adott.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Színes gyöngnyakláncok (2011-DE-14)

A kreatív hódasszony, Judit, gyermekei nyakláncot fűznek. Különböző alakú gyöngyeik vannak (négyzet alakú és kör alakú), melyek kék vagy piros színűek.

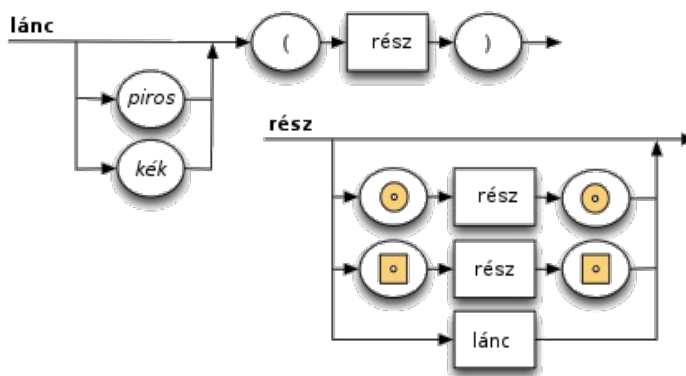
Így például a következő láncot fűzhetik:



Judit elmagyarázza a gyerekeinek, hogy ez a lánc a következőképpen írható le:

piros((*kék* ())

Judit két rajzot készít: ezeket "lánc"-nak és "rész"-nek hívja.



Csak olyan láncokat szeretne, melyek leírása a nyilakat követve megvalósítható.

A kis hódok négy láncot is készítettek, de sajnos csak egy felel meg Judit rajzának. Melyik?



A.



B.



C.



D.

„D” válasz a helyes:

A jelölések a láncrészek beágyazott leírásait eredményezik. Egy láncrész mindig ugyanolyan típusú gyöggel kezdődik és végződik, melyek ugyanolyan színűek is. Minden lánc, amelyik a leírásnak megfelel, felosztható két részre, melyek egymásnak tükörképei. Ez csak a D válaszra igaz.

Az „A” válaszban a színek nem tükörképei egymásnak. A „B” válasznál a lánc közepén a kék kör és négyzet nem tükrösek. A „C” válasz esetében a középén található kék körnek nincs párja, így tükörképe sincs.

Ez informatika!

Judit hód jelöléseit az informatikában Szintaxis diagrammnak nevezzük. Egy programozási nyelv nyelvi szabályai ilyen szintaxis diagrammokkal kerül leírásra, meghatározásra.

Linkek:

http://en.wikipedia.org/wiki/Syntax_diagram



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Csoportmunka (2012-DE-10)

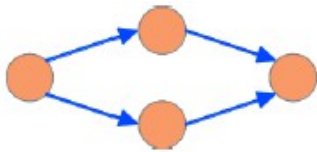
A csoportmunkához a diákok egy osztályban négy csoportot alkottak. Minden csoport felosztotta a munkáját egyéni feladatokra. Három csoport teljesítette az összes feladatát, de a negyedik csoport nem készült el. Mi történt?

A legszorgalmasabb diákok, Anna és Csaba megvizsgálták a négy csoportot. Arra jöttek rá, hogy a legtöbb csapattagnak valakire várnia kellett, mielőtt belekezdhetett volna saját feladatába.

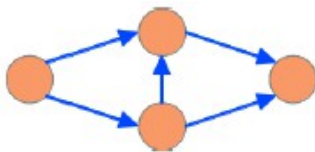
Anna és Csaba minden csoportról rajzolt egy vázlatot, amely a lényegre koncentrált: egy kör egy személyt jelöl. Egy nyíl az 1-es személytől a 2-ig azt jelenti, hogy az 1. személynek be kell fejeznie a munkáját, mielőtt a 2. elkezdhetné.

Melyik kép mutatja azt a csoportot, amelyik nem lett kész a feladataival?

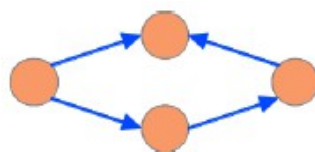
A.



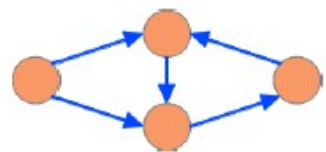
B.



C.



D.



„D” válasz a helyes:

A vázlatok a négy csoport feladatainak függőségi gráfját mutatják. A csoport tagjai nem tudnak továbbhaladni, ha egy ciklus (körút) áll elő. Ekkor senki sem tudja elkezdni a feladatát, mert egy másik feladat befejezésére vár. Csak a D megoldás tartalmaz ilyen ciklust, kört.

Ez informatika!

A legtöbb számítógépes rendszer a különböző feladatokat látszólag egymás után hajtja végre. Egy laptop egyszerre le tud játszani zenét, e-mail-eket fogadni és a háttértárat megtisztítani a vírusoktól. Egy okostelefonon lehet játszani és közben ellenőrizni az SMS-eket és a telefonhívásokat.

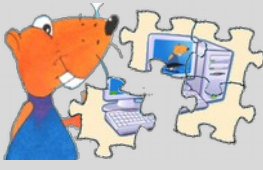
Néhány ilyen „folyamat” viszont függhet egymástól. Amikor egy dokumentumot megnyitunk, a szerkesztő folyamatnak várnia kell, amíg az indítófolyamat a megfelelő adatokat a háttértárra át nem másolta. Mi történik viszont, amikor egyidejűleg egy másik folyamat is elindul a háttértárban? Udvariasan várni fog a két program egymásra, míg a másik az adatait betölti?

Az informatika intenzíven kutatta, hogyan lehetne az ilyen „patthelyzeteket” elkerülni. Aki programozik, annak biztosítania kell, hogy soha ne kelljen két vagy több folyamatnak egymásra várnia.

Linkek:

<http://aries.ektf.hu/~hz/pdf-tamop/pdf-03/html/ch04.html>

http://en.wikipedia.org/wiki/Dining_philosophers_problem



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

A névtelenítéstelenítés (2012-FR-04)

A betegkartonok érzékeny személyes adatokat tartalmaznak, melyek nem kerülhetnek nyilvánosságra. Egy kórház kutatási célokra ezért névtelenül tette közzé az adatokat. A táblázat bal oldala a lista kimenetét mutatja.

Ezzel egy időben – közeledő választások miatt – a kormány az 18250 irányítószámmal rendelkezőkről egy választói listát tett közzé. A táblázat jobb oldala tartalmazza az adatokat minden ilyen január 1-jén született választóról.

születési dátum	nem	irányítószám	betegség	születési dátum	nem	név
1974. 01. 01.	férfi	29400	Diabétesz	1958. 01. 01.	nő	Makk Melánia
1976. 01. 01.	férfi	18250	Tüdőrák	1976. 01. 01.	férfi	Szabó György
1976. 01. 01.	nő	29400	Mellrák	1976. 01. 01.	férfi	Sas Róbert
1976. 01. 01.	nő	29400	Influenza	1984. 01. 01.	nő	Szabad Katalin
1984. 01. 01.	nő	18250	Szívinfarktus	1984. 01. 01.	nő	Müller Éva
1985. 01. 01.	nő	16300	Mellrák	1988. 01. 01.	nő	Bab Ágnes
1987. 01. 01.	nő	25340	Bőrrák	1988. 01. 01.	férfi	Szép Ernő
1988. 01. 01.	férfi	18250	Diabétesz	1988. 01. 01.	nő	Kovács Ica
1988. 01. 01.	nő	18250	Influenza	1989. 01. 01.	férfi	Kakukk Márton

A két táblázat segítségével az egyik választót be tudod azonosítani (deanonimizálni), aki teljesen bizonyosan beteg.

Hogy hívják ezt a beteget?

- A. Szabó György
- B. Szabad Katalin
- C. Szép Ernő
- D. Kovács Ica

„C” válasz a helyes:

Az 1., 3., 4., 6. és 7. sorban lévő betegek nem lehettek, mivel nekik nem 18250 az irányítószámuk. A 2. sorban levő páciens 1976-ban született, férfi és az irányítószáma 18250, viszont két lakos is van, aki megfelel ezeknek az adatoknak: Szabó György és Sas Róbert. Az 5. sorban levő beteg 1984-ben született, nő és az ő irányítószáma is 18250, viszont itt is van két lakos, akiknél ezek az adatok megegyeznek: Szabad Katalin és Müller Éva. A 9. sorban levő beteg 1988-ban született, nő és az ő irányítószáma is 18250 viszont itt is két lakos van, akiknél ezek az adatok megegyeznek: Bab Ágnes és Kovács Ica. A 8. sorban levő beteg viszont 1988-ban született, férfi, az irányítószáma 18250; ez egyedül Szép Ernő lehet.

Ez informatika!

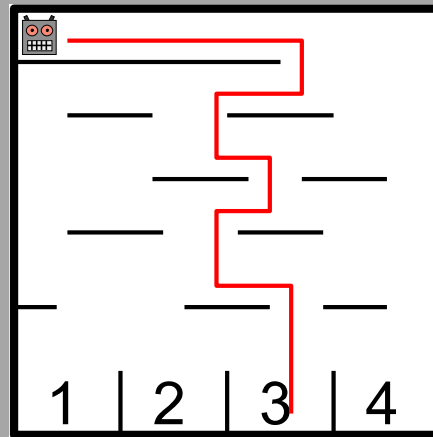
Amikor valakiről adatokat gyűjtenek, akkor ezek alapján egyértelműen újra fel lehet ismerni az embereket. Vagy nem: ekkor emberünk névtelen marad. Hogy valaki névtelen marad, nagyban függ az összegyűjtött adatoktól és az emberektől és az embereknek feltehető kérdések mennyiségétől.

Például ha valakiről csak annyit tudunk, hogy „Müller” a neve és „kb. 85kg” a testtömege, akkor egy világvárosban biztosan névtelen maradna, mivel rengeteg Müller nevű ember van, aki 85 kilós, csak egy lakóházban akár négyen is lehetnek.

Ha valaki ki tudja milyen célból, ezeket a egymástól független információkat összerakja, akkor az itt található „névtelen emberek” anonimitása veszélybe kerül. Ha több és specifikusabb adatokkal lennének a személyek jelölve, a kiválasztottak mennyisége lecsökkenne

Ennek az ellenszere lehetne az adatgyűjtemények kivétel nélküli titkosítása. Ebben az esetben az ilyen jellegű adatok összevetése szembemegy az adatok gyűjtésének eredeti céljával és alapvetően tiltott is lenne.

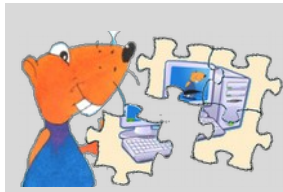
„C” válasz a helyes:



Ez informatika!

A robot egy egyszerű előírást követ, ami az útját a célig meghatározza. Ezeket az előírásokat – vagy az előírások követését – az informatikában algoritmusoknak hívják.

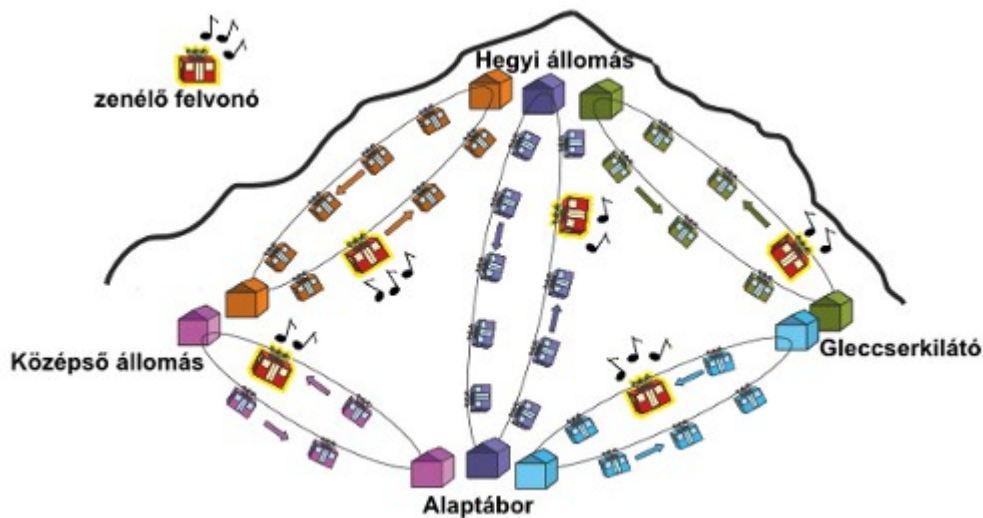
Ezek az algoritmusok nem mindig olyan egyszerűek, mint itt, néha egész bonyolultak is lehetnek. Általában mégsem bonyolult algoritmusokkal lehet bonyolult problémákat megoldani. Épp ellenkezőleg, az informatikában különösen elegáns egyszerű algoritmusokat bonyolult problémákra alkalmazni. Az algoritmusok kitalálása és programozása az informatikusok egyik legfontosabb képessége.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Hegyhódító (2014-AT-03)

Tom szeretne a hegyen lévő felvonóállomáshoz menni. Az alaptábornál különféle felvonókkal tud a hegyi állomáshoz eljutni, viszont ő olyan felvonóval szeretne utazni, amiben zene szól. A kép a felvonók aktuális helyzetét mutatja és Tom az alaptábornál indul.



Minden felvonó az óramutató járásával ellenkező irányba halad. Az utazás időegysége az egyik gondolától az előtte lévő gondoláig az az időtartam, ami a felvonónak kell, hogy az aktuális helyzetéből az előtte lévő helyzetéig eljusson.

Az utazási idő minden felvonónál ugyanaz és akkor sem változik, amikor a felvonók az állomásokon áthaladnak. Az összes felvonópálya minden felvonója mindig egyszerre halad át az állomásokon. Amikor Tom egy állomáson áthalad, idővesztés nélkül át tud ugrani egy másik felvonópálya felvonójára, bár néha várnia kell a következő zenélő felvonóra.

Milyen úton jut Tom a leggyorsabban a hegyi állomáshoz?

- A. Alaptábor → Középső állomás → Hegyi állomás
- B. Alaptábor → Középső állomás → Alaptábor → Hegyi állomás
- C. Alaptábor → Hegyi állomás
- D. Alaptábor → Gleccserkilátó → Hegyi állomás

„D” válasz a helyes:

GA jelölje a zenélő gondolát a Gleccserkilátó-Alaptábor szakaszon.

GH jelölje a zenélő gondolát a Gleccserkilátó-Hegyi állomás szakaszon.

Tomnak 8 időegységre van szüksége. A lemenetel így történik:

1: GA eléri az Alaptábort. Tom beszáll. GH a Hegyi állomás irányába megy.

2: GA elhagyja az Alaptábort. GH eléri a Hegyi állomást.

3: GA a Gleccserkilátó irányába halad. GH elhagyja a Hegyi állomást.

4: GA eléri a Gleccserkilátót. Tom kiszáll. GH a Gleccserkilátó irányába halad.

5: GH eléri a Gleccserkilátót. Tom beszáll.

6: GH elhagyja a Gleccserkilátót.

7: GH a Hegyi állomás irányába halad.

8: GH eléri a Hegyi állomást. Tom kiszáll.

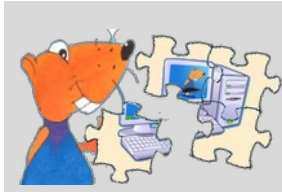
Az A válasznál Tom 10 időegységnyit utazik, a B és C válasz esetében 11-et.

Ez informatika!

Az informatikában ez tűnik a legegyszerűbbnek: a számítógép végrehajtja a a program parancsait szép sorban, egymás után. Más folyamatok számára, amikkel a számítógépek modelleznek és végrehajtananak, a szigorú egymásutánosság nem jó ötlet.

Például egy felvonópálya számára, mert annak mindig csak egy darabkát lehet mennie, mielőtt ismét egy másik felvonópálya kerül sorra az utazáshoz.

Az informatikában ezeknek a felvonóknak a „folyamatok” felelnek meg: egy időben futnak és olykor-olykor „összehangolásra” van szükségük. A felvonók leginkább úgy működnek, mint programrészek, mint egy adatfolyam-architektúra: minden egyes programrészt végre lehet hajtani, ha a szükséges feladatok rendelkezésre állnak. Minden felvonó el tudja vinni Tomot Hegyi állomásra, amikor Tom és egy zenélő felvonó az indulóállomásnál van.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Szülinapi torta (2014-AU-01)

Beatrix az előző születésnapjára egy tortát akart sütni. A torta receptjében 8 fűszer is volt, de miután megsütötte a torta méregzöld színű lett. A vendégek megrémültek, amikor meglátták. De mert nagyon finom volt, Beatrix újra meg akarja sütni. Elhatározta: a torta most már biztosan nem lesz méregzöld. Beatrix úgy sejtí, hogy csak egyetlenegy fűszertől lett a torta méregzöld. Szisztematikusan végignézi, hogy melyik fűszer okozhatta a problémát. Eszébe jut, hogy több próbatortát süssön és közben több fűszerrel kísérletezzon.



Hány tortát kell sütnie Beatrixnak legalább, hogy teljesen biztosan meg tudja állapítani, melyik fűszer okozta a „problémát”?

- A. 2 tortát
- B. 3 tortát
- C. 4 tortát
- D. 5 tortát

„B” válasz a helyes:

A 8 fűszert Beatrix a 3 próbatortán a következőképp osztotta fel:

Próbatorták: 1 2 1, 2 3 1, 3 2, 3 1, 2, 3

Fűszer: 0, 1, 2, 3, 4, 5, 6, 7

A 0-ás számú fűszert egyik próbatortán sem lehet megtalálni, az 1. fűszert csak az 1. próbatortán, a 2. fűszert csak a 2. próbatortán, a 3. fűszert az 1. és 2. próbatortán, stb. Ezzel Beatrix minden fűszert pontosan egyszer kombinált a próbatortákon. Beatrix már a harmadik próbálkozásra tudta, melyik próbatorták lesznek méregzöldek. Ezután már meg tudja határozni a problémás fűszert. Beatrix nem tud rájönni a megoldásra, ha háromnál kevesebbszer próbálkozik. Két próbatortánál csak négy kombináció létezik (--; 1; 2; 1; 2), ebből nem lehet egyértelműen egymáshoz rendelni a 8 fűszert.

Ez informatika!

A kész tortákban információ van tárolva, Beatrixot viszont csak két érték érdekli: a “méregzöld szín” és a “nem méregzöld szín”. Beatrix kísérletében minden próbatorta olyan, mint egy bit a számítógépen. Egy bitben pontosan egy vagy két érték lehet elmentve. Egy bit vagy “bent van” vagy “kint van”. Az informatikában ezeket az értékeket gyakran 1-es vagy 0-ás számjeggyel jelölik. Minél több bit áll a rendelkezésedre, annál több számot tud megjeleníteni – a bináris rendszer segítségével. Három bit megegyezik a fönti példával:

Bitek 000 001 010 011 100 101 110 111

Szám 0 1 2 3 4 5 6 7

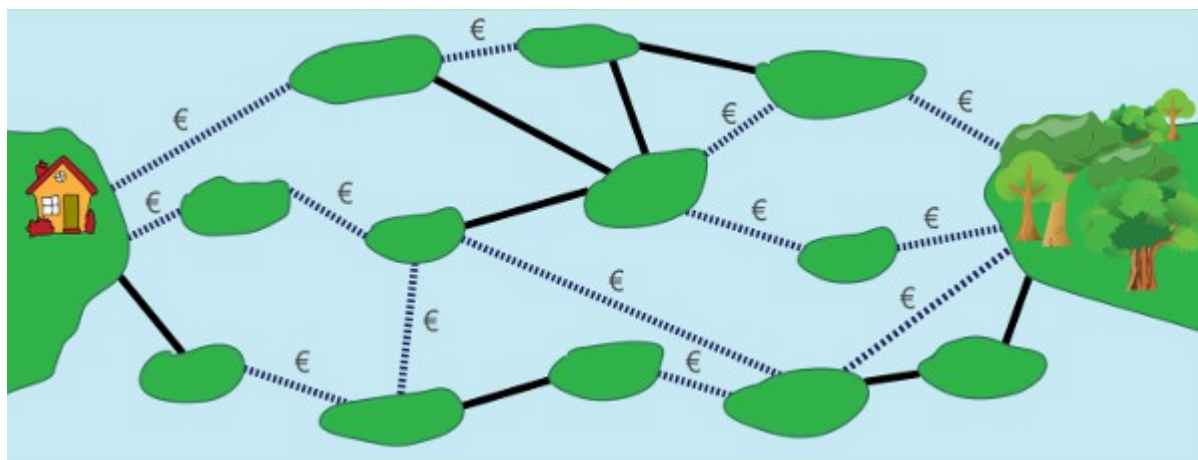
Több bittel nagyobb számokat lehet előállítani, sőt bitekkel egészen más dolgokat is elő lehet állítani pl. betűket. Ehhez ezeket a dolgokat bizonyos számokhoz kell rendelni, amiket bitekkel újra elő lehet állítani. Bitekkel elő lehet állítani majdnem mindent, amit csak el tudsz képzelni. De egyvalamit biztosan nem: a végtelenséget. Ezért minden számítógépes rendszerben, mindegy mekkora, mindig véges a bitek száma.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Drága hidak (2014-CA-04)

A tengeren lévő szigeteket nyilvános és magánhidak kötik össze. A magánhidakon (szaggatott vonal) való átkelés pénzbe kerül. Egy nyilvános hídon (teljes vonal) való átkelésért nem kell fizetni. Sára szeretne a házától az erdőig eljutni. Szeretne a lehető legkevesebb hídon átmenni, de sajnos nincs sok pénze és maximum két magánhídon tud átmenni.



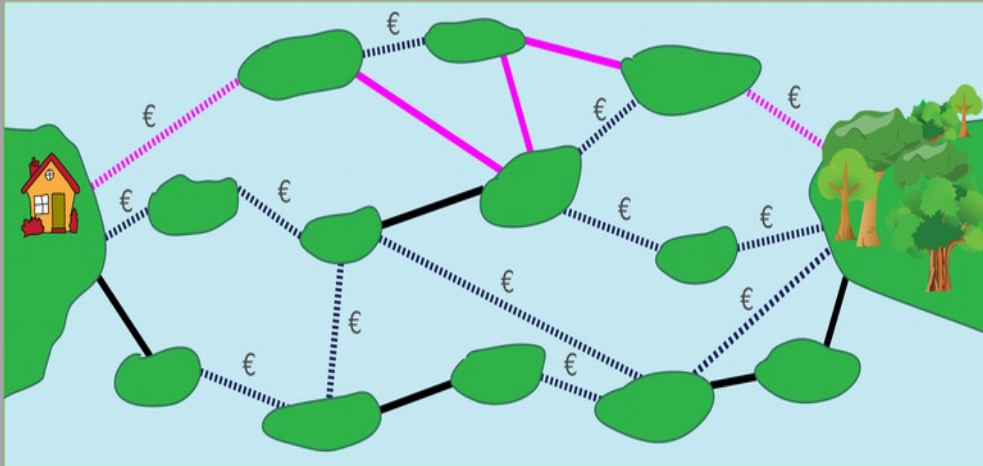
Találd meg azt legrövidebb utat, ami legfeljebb két magánhídon megy át!

Hány híd érint ez az út?

- A. 4 híd
- B. 5 híd
- C. 6 híd
- D. 7 híd

„B” válasz a helyes:

Nem létezik olyan út Sára háza és az erdő között, ami négynél kevesebb hidat érintene. Minden ilyen négy hídon keresztül vezető út három magánhidat érint, szóval Sára nem választhatja ezeket. Az ábra egy olyan öthidas utat mutat, amiből kettő magánhid; ez a legrövidebb út, amit választhat.

**Ez informatika!**

Hidak a szigetek között, utcák a helységek között, hálózati kapcsolatok a számítógépek között, nyomok a platina forrasztási pontjainál: az életben sokféle, látszólag teljesen különböző terület van, ahol tárgyak egymással valamilyen módon összekapcsolódnak.

Ahhoz, hogy olyan hálózatokat építsünk, amik ezeken a területeken használhatóak, az informatika nagyon gyakran egy matematikai modellhez nyúl vissza: a gráfhoz.

Eredetileg a gráfelmélet a svájci zseni, Leonhard Euler Königsbergi hídjából származik. Euler megmutatta 1736-ban, hogy egy körút az éppen akkor létező hidakon Königsberg városában (a mai Kalinyingrád) nem lehetséges. Ő biztosan gyorsabban megtalálta volna Sára útját is.

	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

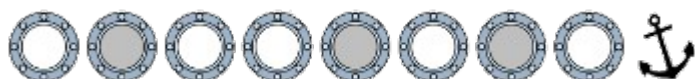
Színezett üveg (2014-CA-05)

Fekete kapitány kicserélteti az üvegeket a jachtjának ablakaiban. Minden új üveg vagy teljesen átlátszó, vagy színezett. Az üveges a következő feladatot kapja:

A hajóablakok a bal oldalon:



A hajóablakok a jobb oldalon:



Mivel két hajóablak mindig pontosan egymással szemben van, minden oldalról keresztül lehet látni a hajón.

Az üvegek színezéstől függően az átláthatóság egészen világos, színezett vagy erősen színezett.

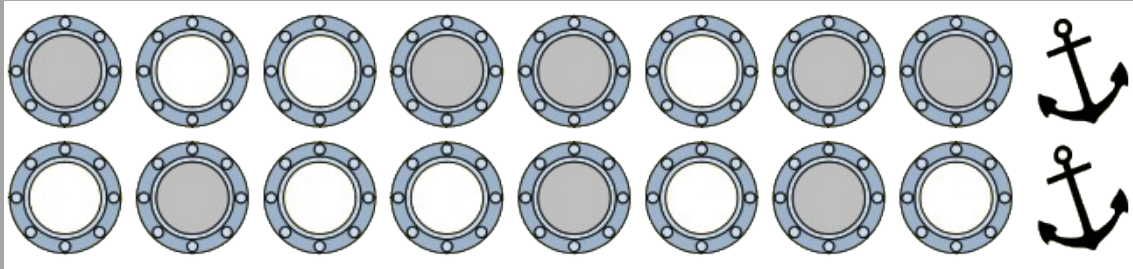


Milyen lesz az ablakok átlátszósága a hajón az üvegezés után?

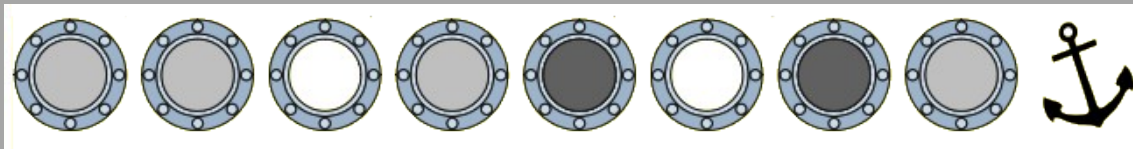


„C” válasz a helyes:

Először azt kell megnéznünk, hogy melyik hajóablak, melyik másikkal van szemben. Ehhez a horgony ad támpontot:



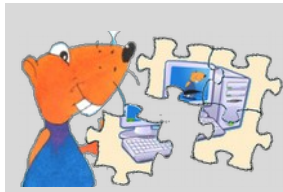
Ha átnézzük ezeken a hajóablakokon, akkor a következő átfedéseket kapjuk:

**Ez informatika!**

Az információ ábrázolása fontos dolog az informatikában. Ebben a feladatban az ablakok szürke árnyalatainak átfedése egy összeadás (pontosabban egy vektor-összeadás – ahol sosem fordul elő átvitel).

Ehhez a teljesen átlátszó ablak a 0, a színezett az 1 és az erősen színezett a 2.

Az összeadás gyors elvégzéséhez az összeadandókat kell egymással összefűznünk, de ehhez fontos, hogy felismerjük: az egyik információ (ablak sor) tükrösen fordított a másikhoz képest.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Hód a gödörben (2014-CA-07)

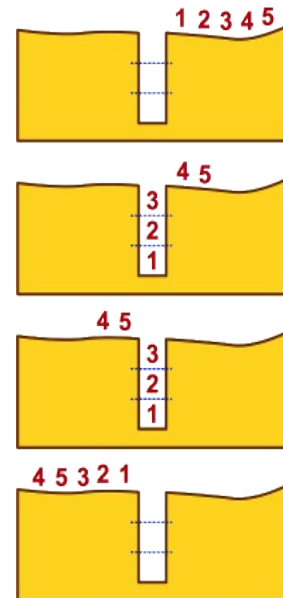
A hódok gyakran mennek csoportokban a sötét erdőn keresztül. Az erdőben az ösvények nagyon keskenyek, ezért is mennek az ösvényeken sorban, egymás után, anélkül, hogy megelőznék egymást. A hódok egy gödörön a következőképpen jutnak át:

Először annyian ugranak be a hódok, amennyien éppen betöltik a gödört.

Ezután a csoport többi tagja 'átmegy' az egész csoporton.

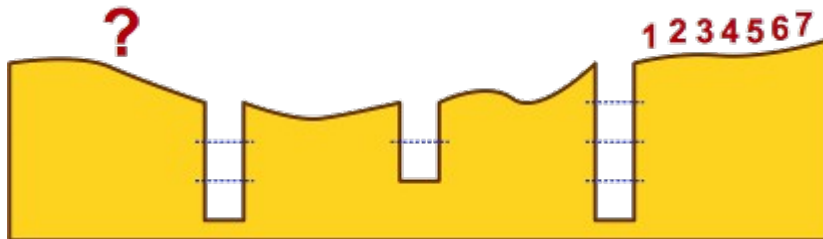
Miután a többi hód áthaladt a hódok kimásznak a gödörből.

Ezután a csoport továbbhalad.



A kép azt mutatja, hogy halad át 5 hód egy gödör fölött. Ebbe a gödörbe 3 hód fér bele.

Egy 7 hódból álló csoport halad keresztül az erdőn. A hódoknak 3 gödörön kell átjutniuk. Az első gödörbe 4 hód, a másodikba 2 és a harmadikba 3 hód fér.



A harmadik gödör után milyen sorrendben haladnak tovább a hódok?

- A. 4 7 5 6 1 2 3
- B. 2 1 6 5 3 4 7
- C. 6 5 7 4 3 2 1
- D. 5 7 6 1 4 3 2

„B” válasz a helyes:

Az elején a hódok sorrendje: 1 2 3 4 5 6 7

Az első gödör után (amibe 4 hód fér): 5 6 7 4 3 2 1

A második gödör után (amibe 2 hód fér): 7 4 3 2 1 6 5

A harmadik gödör után (amibe 3 hód fér): 2 1 6 5 3 4 7

Ez informatika!

Az adatok strukturált elrendezése is nagyon fontos az informatikában, így a tárolt adatokat pontosan vissza lehet hívni. Rengetegféle adatot lehet tárolni és a különbözőféle felhasználásoknál mindig a megfelelőt kell választani.

Egy gyakran használt struktúra volt felfedezhető a példában: a verem (angolul: stack). Ha valamilyen adatot helyezünk el a veremben, akkor az eltakarja a korábban elhelyezett információt és mindig csak az utoljára elhelyezett információt lehet előhívni. Az ilyen rendszereket LIFO-rendszernek hívjuk, a rövidítés (Last – In – First – Out, utolsó – bent – első –kint).

	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Fatörzsképek (2014-CH-01)

Amikor a hódok fatörzseket darabolnak, azokat különösen művészi és ravasz módon fektetik le.

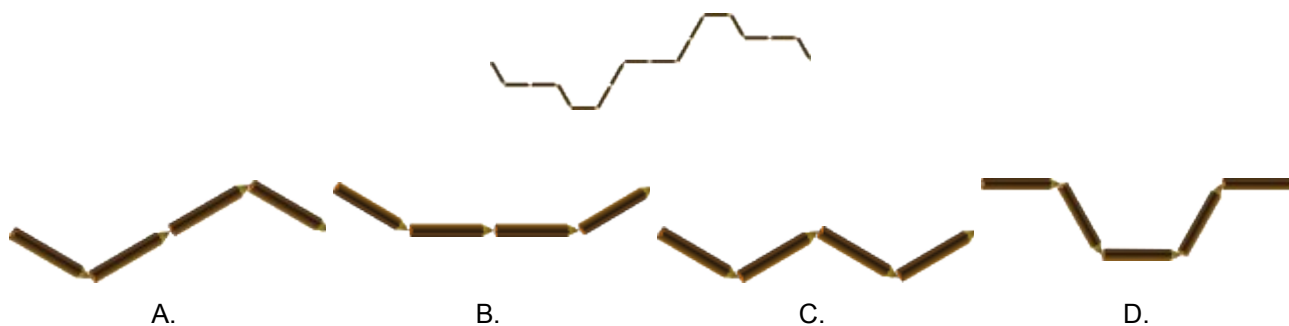
Először egy nagy fatörzset fektetnek le.

Ezt azután egy meghatározott módon kisebb fatörzsekre cserélik. Ezeket a kisebb fatörzseket ugyanazon a módon ismét még kisebb fatörzsekre cserélik.

Itt látható három különböző példa:

Kezdetben	1. csere	2. csere
		
		
		

Ha a második csere után így nézett ki, hogy nézett ki az első csere után?



„A” válasz a helyes:

A többi válasznál így néznének ki az első lépések:

**Ez informatika!**

Ennek a módszernek a helyettesítési eredményét fraktáloknak hívjuk. A fraktálok különleges tulajdonsága, hogy saját magukhoz hasonlóak függetlenül a lépésméretől. Ennek a példafeladatnak az ellentétéként egy fraktál végtelen sok helyettesítésből áll elő. Minél alaposabban szemügyre vesszük, annál több helyettesítést figyelhetünk meg – nincs vége. Egyszerű szabályokkal, mint a helyettesítés, egy elképesztően összetett eredményt érhetünk el.

Informatikában ezt a szabályt szívesen alkalmazzák, mivel kevés programozói ráfordítással igazán sokat elérhetünk. De az igazi, végtelen fraktálokat nem érhetjük el, mivel egy program sem tud végtelenül futni. De ez csak egy elméleti probléma: a gyakorlatban elég, ha a helyettesítéseket olyan gyakran végrehajtjuk, amíg az emberi felhasználó már nem ismeri fel a különbséget a helyettesítési szintek között.

A feladat középső példája egy igen ismert fraktál, a kitalálójáról Koch-görbének nevezik.

Linkek:

<https://hu.wikipedia.org/wiki/Frakt%C3%A1l>

<https://hu.wikipedia.org/wiki/Kateg%C3%B3ria:Frakt%C3%A1lok>

<https://hu.wikipedia.org/wiki/Koch-g%C3%B6rbe>



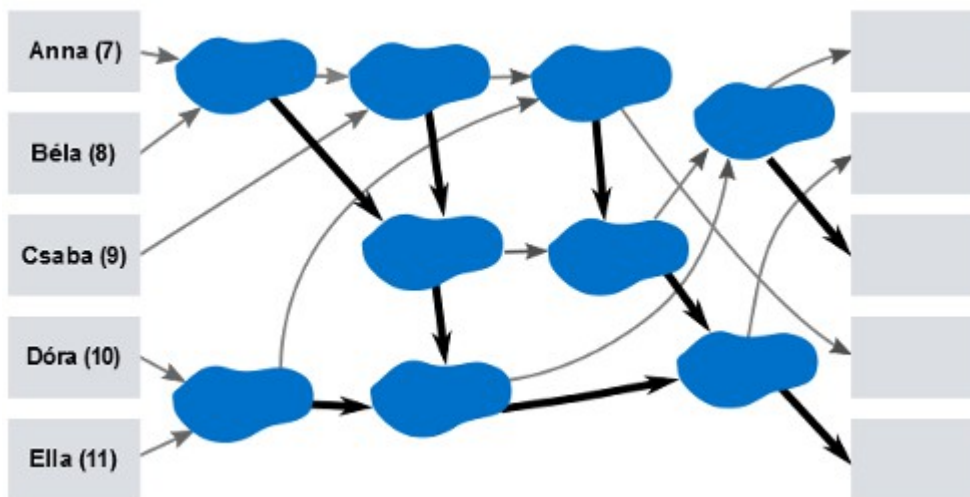
benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Pocsolyaugrálás (2014-CH-02)

Anna (7 éves), Béla (8 éves), Csaba (9 éves), Dóra (10 éves) és Ella (11 éves) egy olyan játékot játszanak, ahol pocsolyáról pocsolyára kell ugrálniuk.

Ehhez nyilakat rajzoltak a földre. Kezdetben a gyerekek a bal oldalon állnak és a nyilak mentén ugrálnak a pocsolyákba.

Az a gyerek, aki először érkezik egy pocsolyába, megvárja, amíg valaki mellé ér. Ezután az idősebb gyerek ugrik a vastagabb, a fiatalabb pedig a vékonyabb nyíl irányába.

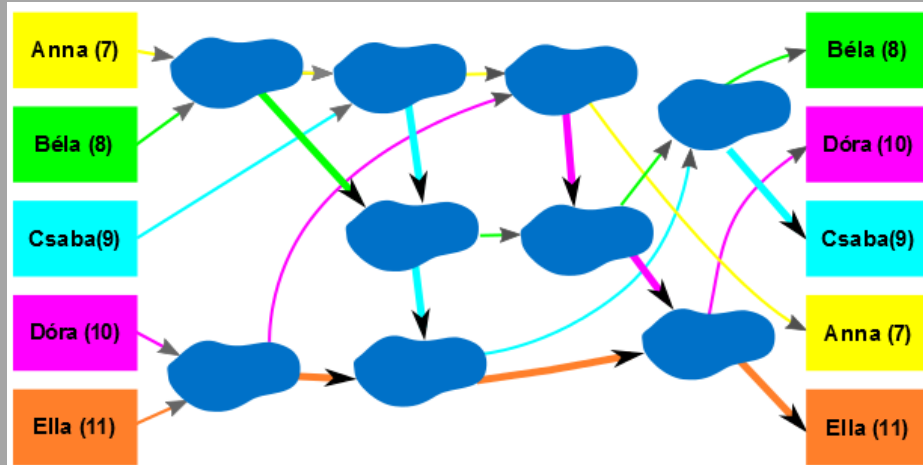


Melyik gyerek lesz a legfelső mezőn?

- A. Anna
- B. Béla
- C. Csaba
- D. Dóra

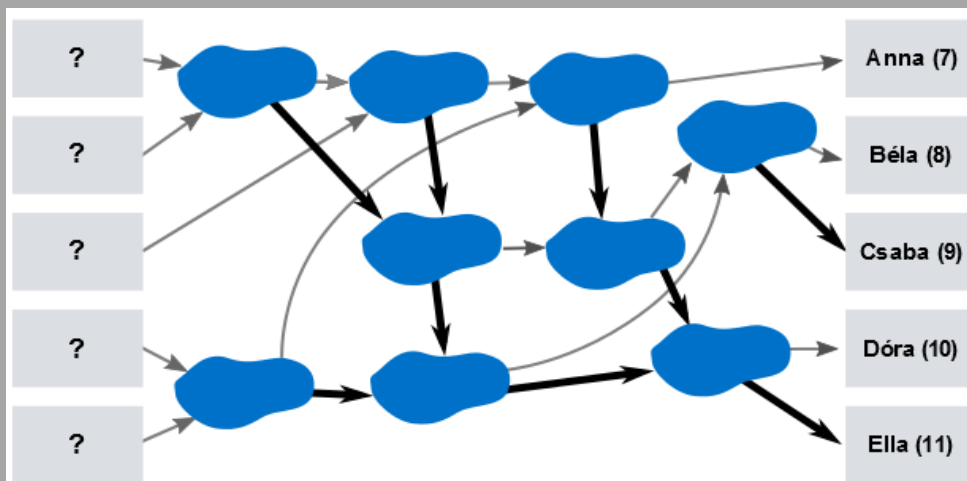
„B” válasz a helyes:

A kép mutatja, hogy ugrálnak a gyerekek:

**Ez informatika!**

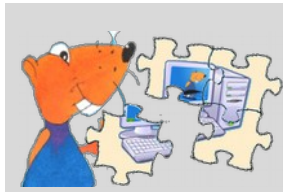
A pocsolyák és a nyilak együtt egy hálózatot alkotnak. A pocsolyák mint hasonlító egységek működnek. Ha a hasonlító elemek jól/helyesen vannak összekötve, akkor a hálózat öt dolgot tetszőleges sorrendbe rendez. Egy ilyen hálózatot rendezőhálózatnak nevezzük.

Mivel a rendezőhálózatokban több hasonlítást párhuzamosan végezhetünk, ezért nagyon hatékonyan rendezhetünk vele. Ebben a feladatban a hálózat nem rendező. A hasonlító elemek nincsenek helyesen összekötve. Az alábbi rajz egy helyesen összekötött rendezőhálózatot mutat:

**Linkek:**

[http://hu.wikipedia.org/wiki/Koml%C3%B3s_J%C3%A1nos_\(matematikus\)](http://hu.wikipedia.org/wiki/Koml%C3%B3s_J%C3%A1nos_(matematikus))

[http://hu.wikipedia.org/wiki/Rendez%C3%A9s_\(programoz%C3%A1s\)](http://hu.wikipedia.org/wiki/Rendez%C3%A9s_(programoz%C3%A1s))

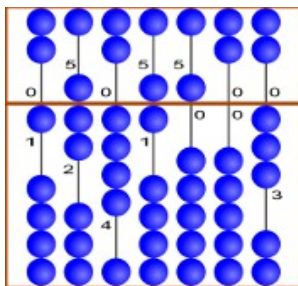
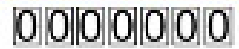
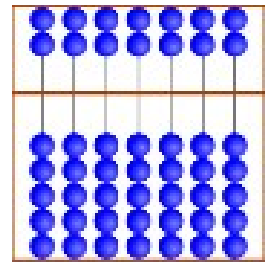


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

suanpan (2014-CH-05)

A „suanpan” egy hagyományos kínai számolótábla. Golyókkal állíthatóak elő a számok. Ehhez a kívánt szám egyes számjegyeit a rudakon kell beállítani.

A felső mezőben minden golyónak 5 az értéke. Az alsó mezőben minden golyónak 1 az értéke. Ha minden rúdon minden golyó a tábla szélén van, akkor a beállított számjegy a 0.



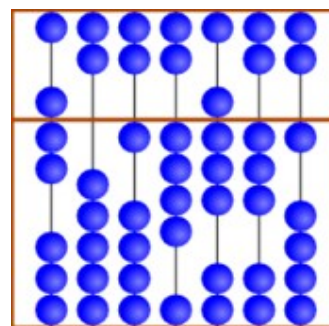
1 7 4 6 5 0 3

Ha egy másik számjegyet szeretnénk előállítani, akkor a kellő számú golyót kell a középvonalhoz tolnunk.

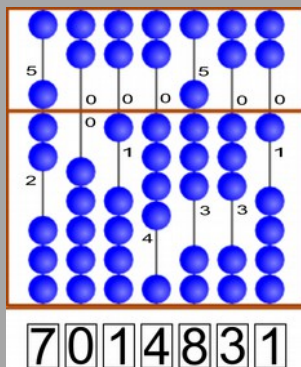
A példában a rudakon az 1, 7, 4, 6, 5, 0 és 3 számjegyek vannak beállítva. Összességében tehát a 1746503 szám van beállítva.

Melyik szám van a képen beállítva?

- A. 7014831
- B. 4763346
- C. 8541224
- D. 3014431



„A” válasz a helyes:



Ez informatika!

Évezredek óta használnak az emberek segédeszközöket ahhoz, hogy nagy számokat megjegyezzenek és azokkal számoljanak. Ebben a feladatban a suanpan-t mutattuk be, az abakusz egy kínai változatát. Suspan-t már hosszú ideje használják és máig sok embernek rendszeres segítség. A suspan a Zhusan számolási módszerrel együtt 2013-ban az UNESCO felvette az „Emberiség szellemi kulturális örökségének reprezentatív listájára”.

Linkek:

<https://hu.wikipedia.org/wiki/Szorob%C3%A1n>

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

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

<http://terebeess.hu/keletkultinfo/szuapan.html>

<http://hirmagazin.sulinet.hu/hu/tudomany/igy-szamoltunk-szuapan-szcsoti-szoroban>



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Valódi négyszög? (2014-CH-07)

Egy robotnak az a szakterülete, hogy négyszöket rajzoljon. Az alábbi egyszerű utasításokat ismeri:

Orange – egy egység hosszú narancssárga vonalat rajzol

Black – egy egység hosszú fekete vonalat rajzol

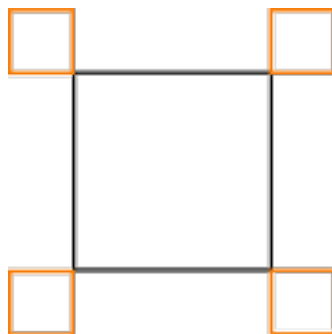
Turn – az óramutató járásának megfelelő irányába fordul 90° -ot

Ezenkívül a robot a következő utasításokat tudja végrehajtani, melyek A és B utasításból állnak, ahol A és B lehetnek egyszerű vagy összetett utasítások:

A, B – végrehajtja A-t és azután B-t

$n \times (B)$ – n-szer végrehajtja B-t.

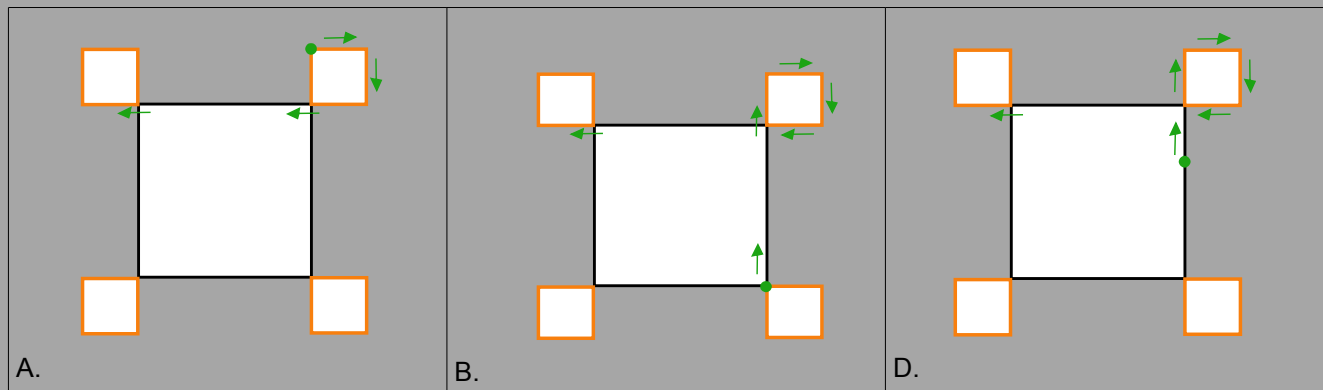
A robotnak a következő ábrát kell előállítania:

**Melyik utasítás NEM a kívánt rajzot állítja elő?**

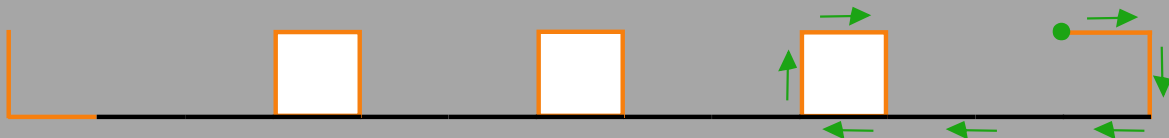
- A. $4 \times (2 \times (\text{Orange}, \text{Turn}), \text{Orange}, 3 \times (\text{Black}), \text{Orange}, \text{Turn})$
- B. $4 \times (3 \times \text{Black}, 3 \times (\text{Orange}, \text{Turn}), \text{Orange})$
- C. $4 \times (2 \times (\text{Orange}, \text{Turn}), 3 \times (\text{Black}), 2 \times (\text{Orange}, \text{Turn}))$
- D. $4 \times (\text{Black}, 3 \times (\text{Orange}, \text{Turn}), \text{Orange}, 2 \times (\text{Black}))$

„C” válasz a helyes:

Az A, B és D utasítás a kívánt rajzot állítja elő:



A C utasítás NEM a kívánt rajzot állítja elő:



Ez informatika!

Akkor is, ha olyan egyszerű programnyelvet használunk, mint a négyszögrajzoló roboté, könnyen követhetünk el hibát. Informatikában a hibát „bug”-nak hívjuk és „debugging” (dibaggolás) a hiba megkeresése a programban.

2014-ben vált ismerté a „Heartbleed bug”. Egy igen elterjedt titkosított adatcserélő programkönyvtár hibájáról van szó. Ez a hiba támadást intézett sok internet-szolgáltatás ellen, mint pl. hozzáférési adatok (felhasználónevek, jelszavak) ellopása.

A bug-oknak kiterjedt hatásai lehetnek. A debugging – a hibák rendszeres elkerülése mellett – az informatika különlegesen fontos munkáihoz tartozik.

Linkek:

[http://hu.wikipedia.org/wiki/Bug_\(informatika\)](http://hu.wikipedia.org/wiki/Bug_(informatika))

<http://xkcd.com/1354/>

	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Csak kilenc gomb (2014-CZ-06)

Dani híreket ír a régi mobiltelefonján.

Minden betűnél a neki megfelelő gombot kell megnyomnia egyszer, kétszer, háromszor vagy négyszer.

Ezután egy rövid szünet következik.

A „C” betűhöz például háromszor nyomja le a gombot a 2-es számmal, mivel a gombon a harmadik betű a C.

A „BOT” szó leírásához összesen hatszor kell gombot nyomnia: kétszer a 2-est, háromszor a 6-ost és egyszer a 8-ast.



Dani hatszor nyom le gombot ahhoz, hogy leírja a barátnője nevét.

Hogy hívják a barátnőjét?

- A. Miriam
- B. Emma
- C. Iris
- D. Ina

„D” válasz a helyes:

„Miriam”-ban ugyan csak hat betű van, de 12-szer kell gombot nyomni: egyszer a 6-ost, háromszor a 4-est, háromszor a 7-est, egyszer a 2-est és egyszer a 6-ost.

„Emma” leírásához csak hat gombnyomás kell: kétszer a 3-as, egyszer a 6-os, ismét egyszer a 6-os és egyszer a 2-es.

„Iris” leírásához 13 gombnyomás kell: háromszor a 4-es, háromszor a 7-es, háromszor a 4-es és négyszer a 7-es.

„Ina” leírásához hat gomblenyomás szükséges: háromszor a 4-es, kétszer a 6-os és egyszer a 2-es.

Ez informatika!

Egy kisebb billentyűzeten csupán kilenc billentyűvel is egyértelműen megadhatjuk az ábécé minden betűjét és még pár írásjelet is. Ehhez az szükséges, hogy a jeleket/karaktereket úgy különböztessük meg, hogy hányszor nyomjuk le a billentyűket. A karakterek így a gombnyomások számával kerülnek kódolásra.

A régebbi mobiltelefonoknál ez a kódolás sokszor volt szükséges, mivel csak kis billentyűzet elhelyezésére volt lehetőség.

Pár éve egyre több mobiltelefon érintőképernyős. Ezzel lehetővé vált, hogy a billentyűzet a képernyőn jelenjen meg. Egy új technológia, mint az érintőképernyő, egy új beviteli lehetőséget tett lehetővé.

Hogy a beviteli lehetőségek és a mobiltelefonok 10 év múlva hol tartanak, nehéz megmondani. De egészen biztos, hogy máshol, mint most. Már most vannak telefonok, melyek képesek a hangfelismerésre.

Linkek:

<http://en.wikipedia.org/wiki/Multi-tap>

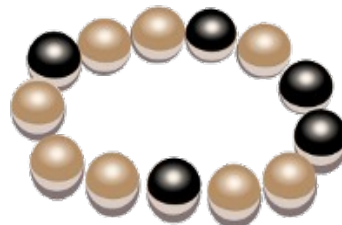


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Hamis karkötők (2014-CZ-08)

A legutóbbi víziparádén a Hódhercegnő viselte a mágikus karkötőjét, amely világos és sötét gyöngyökből áll. Ezután egy dobozkába tette.

Most ismét szüksége van a mágikus karkötőre és kinyitja a dobozkát. De jaj, valaki három hamis karkötőt tett az igazi mellé.



A négy karkötő közül melyik lehet az eredeti mágikus?

A.



B.



C.



D.



„B” válasz a helyes:

A mágikus karkötő 13 gyöngyből áll. Ebből 5 sötét. A sötétek közül kettő egymás mellett van.

Az A karkötő hamis, mivel nincs két sötét gyöngy egymás mellett.

A C karkötő hamis, mivel csak 12 gyöngyből áll.

A D karkötő hamis, mivel 6 sötét gyöngy van benne.

Ez informatika!

A karkötők példák tárgyak strukturált elrendezésére. Az informatika mintákról (pattern) beszél.

A minták megfelelő szerkezeti tulajdonságaikban hasonlóság vagy megfelelés (minta azonosság) alapján egymással összehasonlíthatóak.

Ebben a feladatban a szerkezeti tulajdonság a gyöngyök száma, a sötét gyöngyök száma és a párosítás.

Az informatika a mintafelismerésről (pattern recognition) beszél, amikor a mintát mint egy nagyobb szerkezet összetevőjét keresi. Ez lehet egy meghatározott szó egy szövegben, vagy egy arc egy megfigyelőkamera felvételén. És még sok más.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Fagylalthalom (2014-CH-02)

A LIFO Fagylaltozóban a kívánt fagylaltgombócokat egy tölcsérre halmozzák. Pontosan abban a sorrendben, melyben a vendég kéri.

Mit kell mondania a vevőnek, ha a képen látható fagylalttölcsért szeretné?

Szeretnék ...

- A. ... egy csokoládét, egy mentát és egy áfonyát.
- B. ... egy csokoládét, egy áfonyát és egy mentát.
- C. ... egy áfonyát, egy mentát és egy csokoládét.
- D. ... egy áfonyát, egy csokoládét és egy mentát.



„C” válasz a helyes:

„Szeretnék egy áfonyát, egy mentát és egy csokoládét!”

Amit először mond, az kerül elsőnek, legalulra a tölcsérbe.

Amit utolsónak mond, az kerül utoljára, legfölülre a halomba.

Az A válaszban a sorrend pont fordított.

A B és a D válaszban nem a menta van középen.

Ez informatika!

A sorrend fontos! Ha a fagyaltféléket más sorrendben mondanánk, egy másik fagyalttölcsér lenne belőle.

Az informatikában megtanuljuk, milyen hasznos, ha valami rendezett.

És hogy meg kell érteni, melyik rendezést melyik helyzetben kell alkalmaznunk. Ennek megértése nélkül, ahogy a fagyalttölcsér működik, nem tudnánk a kívánt fagyalttölcsért megrendelni. A helyzet megértése nélkül nem tudunk kellő programot készíteni.

Ebben a feladatban a használt rendezés azt jelenti: „last in, first out” (LIFO, azaz utolsónak be, elsőnek ki)

Linkek:

[http://hu.wikipedia.org/wiki/Verem_\(adatszerkezet\)](http://hu.wikipedia.org/wiki/Verem_(adatszerkezet))

<http://hu.wikipedia.org/wiki/Fagyalt>

	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Lábnymok (2014-DE-02)

Lábnym-fák! Egy bizonyos rendszer alapján kerülnek létrehozásra.

Ez a lábnymprogram 1-fára:

Menj egy lépést előre, ezzel elkészítesz egy lábnymot.
Lépj ismét vissza



Ha már ismerjük az 1-fa programot, akkor így néz ki a 2-fa lábnymprogram:

Menj két lépést előre, így két lábnymot készítesz.
Fordulj jobbra és készíts egy 1-fát
Fordulj balra és készíts egy 1-fát
Menj vissza a nyomodon.



A 3-fa lábnymprogram is könnyen elmagyarázható, ha ismerjük a 2-fa programot:

Menj 3 lépést előre, ezzel elkészítesz három lábnymot
Fordulj jobbra és készíts egy 2-fát
Fordulj balra és készíts egy 2-fát
Menj vissza a nyomodon.



A 4-fa lábnymprogram ugyanezzel a módszerrel leírható.

Ennek alapján melyik fa (rajz) egy 4-fa?



„A” válasz a helyes:

Ha összehasonlítjuk a 2-fa és a 3-fa programokat, akkor felismerjük a rendszert és felírhatjuk a 4-fa programot:

```
Menj 4 lépést előre, ezzel elkészítesz 4 lábnymot
Fordulj jobbra és készíts egy 3-fát
Fordulj balra és készíts egy 3-fát
Menj vissza a nyomodon.
```

Csak az A ábra állhat elő ebből a programból. Ezért ez a 4-fa lábnymprogram, hiszen 4 lábnymból és két 3-fából áll.

A B válaszban 4 lábnym van, de három 3-fa.

A C válasz ugyan két 3-fát tartalmaz, de csak három lábnymot.

A D válasz négy lábnymmal kezdődik, de az alfák nem 3-fák.

Ez informatika!

A rendszer az összes lehetséges n -fára működik. N -fa lábnymprogram azt jelenti, hogy n lépést megyünk előre, ehhez n lábnymot kell hagynunk, majd két $(n-1)$ -fát kell készítenünk, majd visszamennünk.

Egy $(n-1)$ -fa viszont $(n-1)$ lábnymból és két $(n-2)$ -fából áll, és így tovább, amíg el nem jutunk az 1-fáig.

Az informatikában rekurzióról beszélünk, amikor egy feladatot kell elintéznünk, melyben ugyanannak a feladatnak egy egyszerűbb változata szerepel, egészen addig, amíg a feladat legegyszerűbb változata egy speciális módon elvégzésre nem kerül. Sok esetben elegáns módon leírhatjuk rekurzióval, hogyan kell megoldani egy feladatot.

De vigyázzunk: egy n -fa esetében 2 $(n-1)$ -fát, 4 $(n-2)$ -fát, 8 $(n-3)$ fát, ... $2^{(n-1)}$ 1-fát kell készítenünk. Amennyiben n nagy szám, ez nagyon sokáig is eltarthat. A rekurzió tehát elegáns, de fáradtságos is.

Linkek:

<https://hu.wikipedia.org/wiki/Rekurzi%C3%B3>

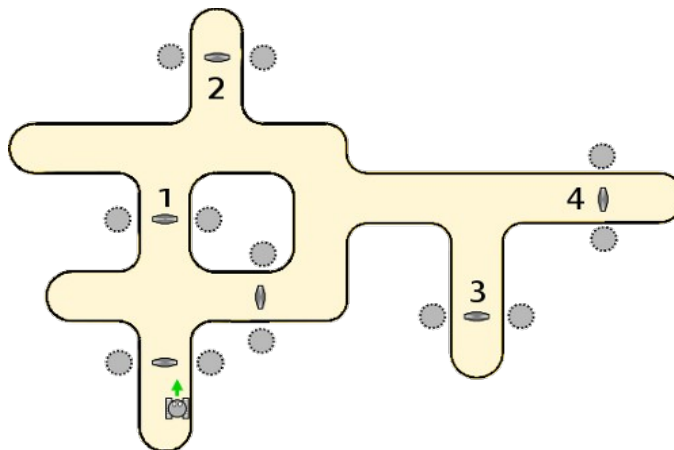
	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Takarító robot (2014-DE-05)

Egy robot mindig a sávjának (az útjának) a szélén halad végig. A robot a következő utasításokat kaphatja és hajtja végre:

Utasítás	Végrehajtás
START-GO	Elindítja a motort és elindul a kezdő irányba.
GO	Továbbmegy a sáv szélén.
CROSS-GO	A sávjának másik szélére vált és ugyanabban az irányban halad tovább.
STOP	Megáll.

Ha a robot áll, az elindításához meg kell kapnia a START-GO utasítást. Az útvonalán jeladók vannak elhelyezve. Mindig, amikor a robot áthalad egy jeladón, a következő parancsot hajtja végre.



A kép mutatja a robot útját a jeladókkal. Alul láthatod a robotot és az indulási irányát. A robot áll.

Aztán megkapja az alábbi parancsokat:

START-GO

CROSS-GO

GO

GO

GO

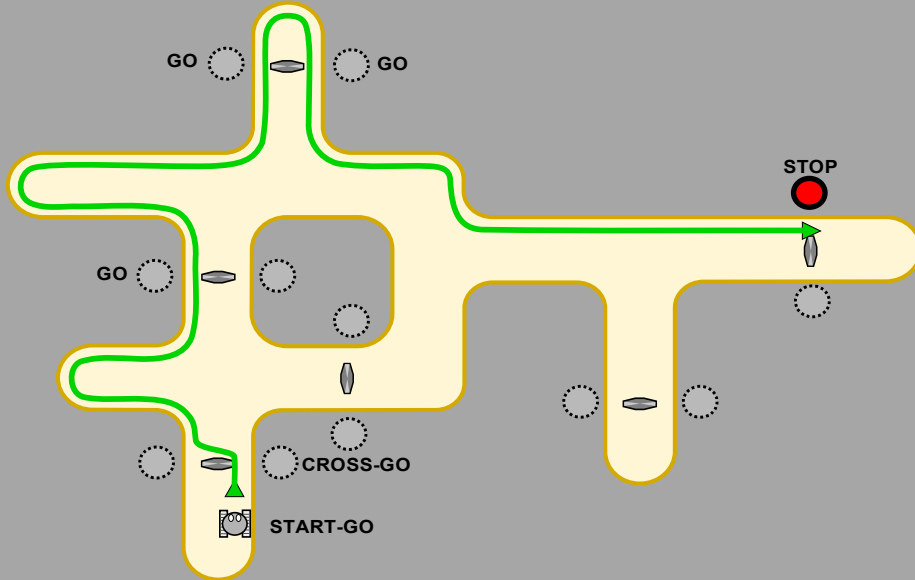
STOP

Melyik ponton áll meg a robot?

- A. 1-es ponton
- B. 2-es ponton
- C. 3-as ponton
- D. 4-es ponton

„D” válasz a helyes:

A robot a 4-essel jelölt ponton áll le. A kép mutatja az útját:



Ez informatika!

Mozgó (helyváltoztató) robotokat (automatikus járművek vezető nélkül) találhatunk reptereken, gyárakban vagy kórházakban. Ezek a gépek programok által vezéreltek. Egyszerű esetben ez a program utasítások sorozata – mint ebben a feladatban. A valós robotok esetében azonban sokkal összetettebb programok is előfordulhatnak.

Az informatikában sok ember dolgozik robotok beprogramozásán: mozgó (helyváltoztató) robotok, építő robotok, orvosi robotok, focizó robotok, robotpilóták, ...

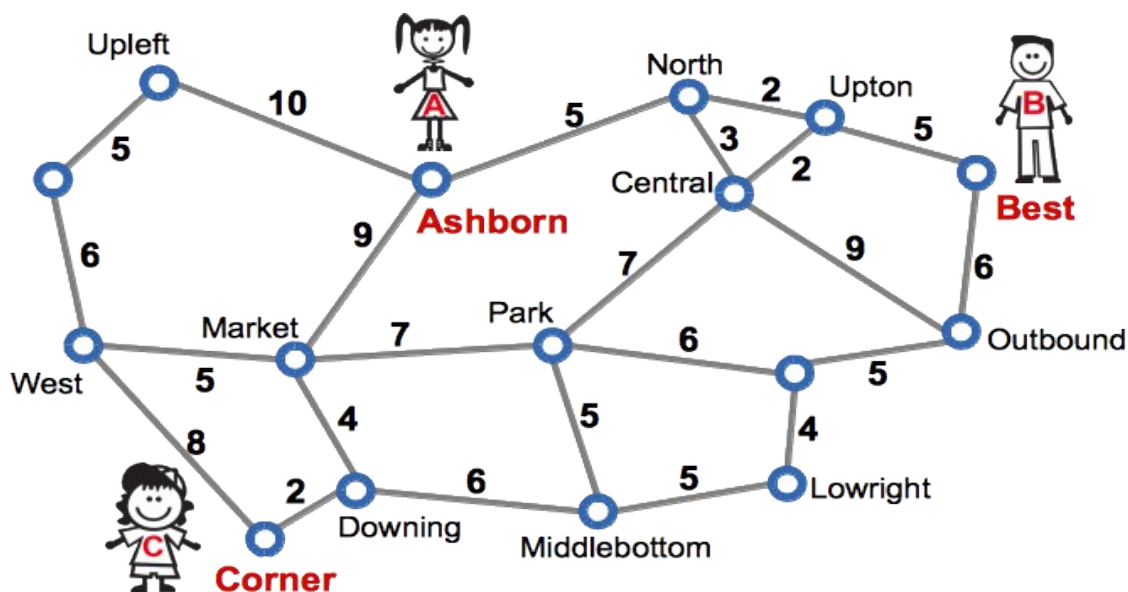
A robotok viselkedése sokszor kihat a környezetünkre és így az emberre is. A robotokat irányító programoknak ezért különösen megbízhatóknak kell lenniük.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Találkozás (2014-DE-07)

Anna, Béla és Cecília egy jól kiépített metróhálózattal rendelkező városban laknak. A metróhálózat rajza (lásd a képet) mutatja a megállókat és a megállók közötti szakaszokat. Minden szakaszra meg van adva, hány perc szükséges a megtételéhez.



Anna az Ashborn megállónál, Béla Bestnél, Cecília pedig Corner megállónál lakik.

Szeretnének valamelyik megállóban találkozni, de mindegyikőjük legfeljebb (maximum) 15 percet szeretne csak utazni.

Az alábbiak közül melyik megállóhely jöhet szóba a találkozóhoz?

- A. Ashborn
- B. Market
- C. Central
- D. Lowright

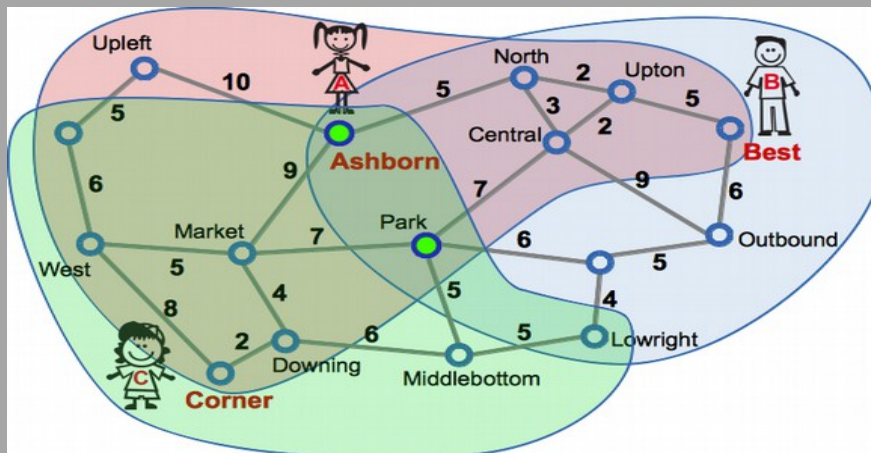
„A” válasz a helyes:

A Park és Ashborn megállók jöhetnek csak szóba. Csak ezt a két megálló érhetik el mindhárman maximum 15 perc alatt, amennyiben a következő szakaszokat használják:

Park: Ashborn-North-Central-Park: 15 perc; Best-Upton-Central-Park: 14 perc; Corner-Downing-Market-Park vagy Corner-Downing-Middlebottom-Park: 13 perc. (a látszólag közvetlen úthoz Ashborn-Market-Park Annának több időre lenne szüksége, 16 percre)

Ashborn: Ashborn-Ashborn: 0 perc (Annának tehát nem kellene utaznia); Best-Upton-Nort-Ashborn: 12 perc; Corner-Downing-Market-Ashborn: 15 perc.

Az alábbi képen láthatóak színesen a területek, melyeken az állomásokat Anna, Béla és Cecília maximum 15 perc alatt elérhetik. Csak Ashborn és Park állomások fekszenek a három terület metszetében. Tehát más találkozási pont nem létezik.

**Ez informatika!**

A csak egy halmaz elemei közti kapcsolat gyakran gráfként kerül kibővítésre: az elemeket csomópontoknak hívjuk és a csomópontok közötti kapcsolatok az élek. Sok gráfban az éleknek irányuk is van: az „a” csomópont kapcsolatban áll „b” csomóponttal, de fordítva nem. Ezen kívül az éleknek meghatározott értéket, úgynevezett „súlyt” is adhatunk.

Egy közlekedési hálózat, mint a feladatban is, nagyon jól modellezhető gráfokkal, élsúlyokkal, melyek a közlekedési időt jelentik. Szerencsére az informatikában sok erős algoritmust fejlesztettek már ki gráfokra, többek között olyat, ami csomópontok között a legrövidebb utat (élek sorozatát) találja meg. A „legrövidebb út algoritmusok”, mint pl. az Edsger W. Dijkstra által meghatározott, az útvonaltervezés, autós navigációs rendszerek alapját képezik.

Linkek:

<http://hu.wikipedia.org/wiki/Dijkstra-algoritmus>

http://hu.wikipedia.org/wiki/Moh%C3%B3B3_algoritmus

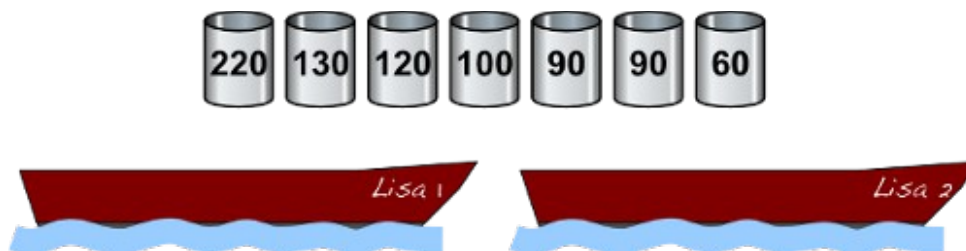


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Lisák feltöltése (2014-DE-08)

Bertalané és Barnabásé, a két halászé a „Lisa1” és „Lisa2” hajó – a két Lisa.

Mindkét hajó legfeljebb 300kg-ig terhelhető.



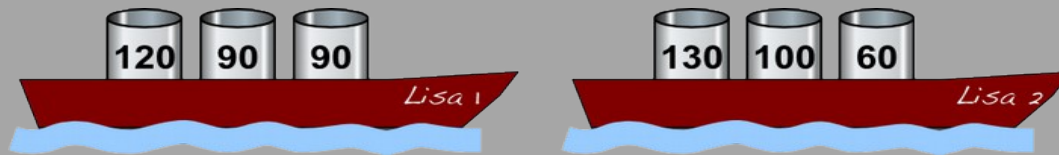
Bertalannak és Barnabásnak a két hajóval el kell szállítania pár hordó halat. A szállítást súly alapján fizetik.

Maximum mennyi halat tudnak egyszerre a két hajóval elszállítani?

- A. 810kg
- B. 600kg
- C. 590kg
- D. 530kg

„C” válasz a helyes:

Összesen 590 kg halat tudnak elszállítani: $120+90+90=300$ az egyik hajón és $130+100+60=290$ kg a másik hajón.



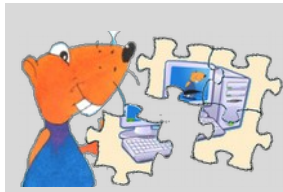
Vigyázz, ne legyél mohó! Ha legelőször a legnehezebb hordókat veszed a hajók megtöltéséhez, maximum $220+60=280$ és $130+120=250$, azaz összesen 530 kg halat tudsz felpakolni.

590 kg-nál több hal nem pakolható fel. Ehhez mindkét hajónak 300 kg-ot kellene szállítania. Nincs azonban csak egy lehetőség ($120+90+90$) az adott hordókból 300 kg-nyi halat összerakni.

Ez informatika!

Sok ember el van ragadtatva attól, hogy dolgokat optimalizáljon – gyakran költséget spóroljon és a bevételét maximálja. A nem egyszerű problémák esetében legtöbbször számítógépes programokat alkalmaznak az optimalizáláshoz: a legrövidebb út, az optimális terhelés, az ideális órarend megtalálásához. Egyes optimalizálási problémák úgynevezett „mohó” algoritmussal oldhatóak meg. Ennél a megoldáshoz vezető minden lépés (itt a hordók kiválasztása) úgy kerül meghatározásra, hogy annyi profitot hozzon, amennyi lehetséges – ettől mohó.

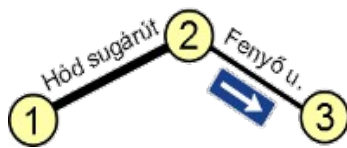
Ami szép az informatikában: a legtöbb esetben a mohóság nem segít többet és összetettebb algoritmusok alkalmazása szükséges az optimális megoldás megtalálásához. Egyes problémák esetében bizonyítható, hogy az olyan algoritmusok, melyek az optimális megoldást megtalálják, magától a számítógéptől elfogadhatatlanul nagy erőfeszítést igényelnek. Sok ilyen nehéz optimalizálási problémára az informatika hatékony algoritmusokat alakított ki, melyek korántsem a legjobb (optimális), de kimutathatóan nagyon jó, a majdnem optimális megoldást találják meg.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

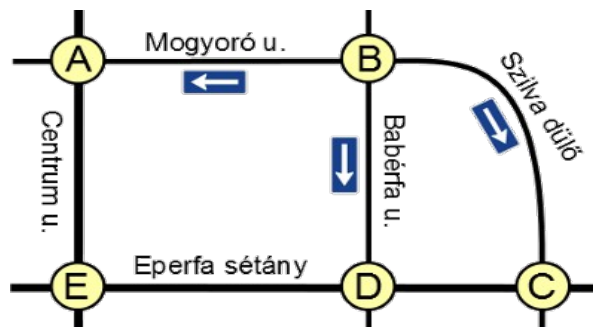
Közlekedés a városban (2014-ES-02)

Hódfürdőn a Fenyő utcát egyirányúsították. Jánosnak, a környék egyetlen taxisának újra meg kell jegyeznie, hogy juthat el egyik helyről a másikra. A három csomópontot (1, 2 és 3) János egy táblázatba tette. Pipákat tett a táblázat mezőibe, hogy megjegyezze, melyik utcában melyik irányban haladhat.



	1	2	3
1		✓	
2	✓		✓
3			

A szomszédos Hódfalvában is egyirányúsítottak pár utcát.



János Hódfalvához is készített egy táblázatot pipákkal.

A térkép alapján melyik János táblázata?

A.

	A	B	C	D	E
A		✓			✓
B	✓		✓	✓	
C		✓		✓	
D	✓		✓		✓
E	✓			✓	

B.

	A	B	C	D	E
A					✓
B	✓		✓		
C				✓	
D			✓		✓
E	✓				

C.

	A	B	C	D	E
A		✓			✓
B	✓			✓	
C		✓		✓	
D			✓		✓
E	✓			✓	

D.

	A	B	C	D	E
A					✓
B	✓		✓	✓	
C				✓	
D			✓		✓
E	✓			✓	

„D” válasz a helyes:

Az X sor Y oszlop mezőjében (röviden: mezőjében(X,Y)) egy pipa azt jelenti, hogy János az X csomópontból az Y csomópontba vezethet. A mindkét irányban járható utcák (mint pl. az Eperfa sétány a D és E csomópontok között) két pipát jelentenek a táblázatban: egyet a D sor E oszlop mezőjében és egyet az E sor D oszlop mezőjében.

Ha két csomópont között egyirányú utca van (mint pl. a Szilva dűlő B-ből C csomópontba), akkor csak egy pipa kerülhet a táblázatba. A mi esetünkben a B sor C oszlopába.

Ez informatika!

A kitöltött táblázat pontosan azt mondja meg, melyik csomópontból, melyikbe utazhatunk. A csomópontok közötti kapcsolatok különleges tulajdonságairól azonban semmit sem mond. Egy taxis valószínűleg azt is szívesen tudná, milyen gyorsan vezethet, van-e dugó az útszakaszon, milyen az útburkolat. De annak eldöntésére, hogy A-ból B-be eljuthatunk-e, esetleg van-e több kapcsolat, elég információt ad a pipák elhelyezése a táblázatban.

Információs rendszerek (és az emberek is) általában csak annyi információt dolgoznak fel, amennyi épp szükséges. A valóság egy elvonatkoztatott modelljét használják.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Hód igazolvány (2014-ES-03)

Minden hódnak van egy igazolványa igazolványszámmal.

Hogy megelőzzék az olvasási hibákat, minden igazolványon van még egy ellenőrző karakter.

Az ellenőrző karakter előállításra a következő:

1. Az igazolványszám számjegyeit összeadjuk.
2. Kikeressük a kapott számot a táblázatból.

A táblázat jobboldali oszlopából kiolvassuk az eredményhez tartozó ellenőrző karaktert.

Eredmény	Ellenőrző karakter
0 7 14 21 28	T
1 8 15 22 29	R
2 9 16 23 30	W
3 10 17 24 31	A
4 11 18 25 32	G
5 12 19 26 33	M
6 13 20 27 34	Y

**Mi a képen látható igazolvány ellenőrző karaktere?**

- A. A
- B. M
- C. Y
- D. T

„A” válasz a helyes:

$$4+5+1+7=17$$

A 17-es szám a táblázat negyedik sorának harmadik száma. A negyedik sor jobboldali cellájában/oszlopában az 'A' karakter található.

Ez informatika!

Számsorok kiolvasására az informatika több módszert és eszközt is kifejlesztett, melyek a mindennapi helyzetekben egy tárgy vagy egy személy „azonosságát” igazolják.

Az azonosság igazolása sok területen fontos lehet. Egy bankjegy értéke vagy egy jótállás, egy koncertjegy, egy repülőjegy érvényessége, autók vagy más járművek rendszáma: ezek mindegyike – és még sok más is – biztosan felismerhető kell legyen.

Számjegyek gépi adatbeolvasásakor azonban olvasási hiba is felléphet. Ha a hibát nem azonnal ismerik fel, később rendkívül bosszantó lehet – akár a vizsgált, akár a vizsgáló, akár mindkettő számára.

Az olvasási hibák felismerésére legelterjedtebb módszer egy azonosító számcsoporthoz egy vagy több ellenőrző szám kiszámolása és a számhoz való illesztése. Olvasási hiba esetén a beolvasott számcsoporthoz az ellenőrző kód nem illik össze.

UI: Autó rendszámok nem tartalmaznak ellenőrző kódot.

Linkek:

http://en.wikipedia.org/wiki/Check_digit



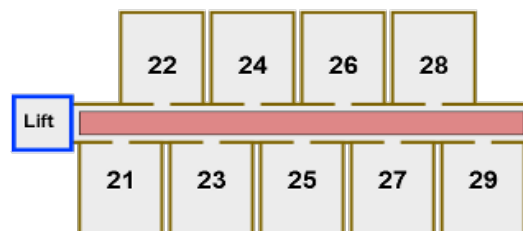
benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Komfort hotel (2014-FI-02)

A Komfort Hotelben a szobaszámok két számjegyűek:

- Az első számjegy az emeletet jelzi, ahol a szoba fekszik;
- A második számjegy azt adja meg, hogy milyen messze fekszik a szoba a felvonótól.

A szobák minden emeleten úgy kerültek elrendezésre, mint a 2. emeleti tervrajzon látható.



A Komfort Hotelben a vendégeknek minél kevesebbet kell fáradniuk. Minél közelebb van egy szoba a lifthez, annál komfortosabb. Ha két szoba azonos távolságban van a lifttől, akkor az alsóbb emeleten lévő szoba a komfortosabb. Tehát a 32-es szoba komfortosabb, mint a 15-ös és a 22-es szoba komfortosabb, mint a 32-es.

A Komfort Hotelben az alábbi előírás létezik: egy új vendég mindig a legkomfortosabb szabad szobát kapja.

Jelenleg a következő szobák szabadok: 12, 25, 11, 43, 22, 15, 18, 31, 44, 52.

Tíz új vendég érkezik egymás után.

Milyen sorrendben adják ki nekik a szobákat?

- 18, 15, 12, 11, 25, 22, 31, 44, 43, 52
- 52, 43, 44, 31, 22, 25, 11, 12, 15, 18
- 11, 31, 12, 22, 52, 43, 44, 15, 25, 18
- 11, 12, 15, 18, 22, 25, 31, 43, 44, 52

„C” válasz a helyes:

A szobák kiadásának sorrendje azt jelenti, hogy a szobaszámokat először a második számjegyük szerint, majd az első számjegy alapján kell sorba állítani.

A szobák helyes sorrendjének megadásához a szobaszámokat jobbról balra kell olvasni és a kisebbik érték szerint rendezni.

Például a 32-ből 23, a 15-ből 51 lesz és mivel $23 < 51$, ezért a 32-es szobát előbb kell kiadni, mint a 15-et.

Ha így olvassuk a számokat, csak a C válasz adja a helyes sorrendet (11, 13, 21, ..., 52, 81).

Az A válasz hamis: az első két szám (18, 15) esetében rossz a sorrend ($81 > 51$).

A B válasz hamis: a harmadik és a negyedik szám (44, 31) esetében rossz a sorrend ($44 > 13$).

A D válasz hamis: a negyedik és az ötödik szám (18, 22) esetében rossz a sorrend ($81 > 22$). Itt először a szintek és azon belül a távolságok alapján kerültek rendezésre a szobák.

Ez informatika!

Komfort Hotel szobáinak sorrendbe rakása speciális rendezés. De van egy különleges tulajdonsága: ha a hagyományosan sorrendbe rakott szobaszámokat – pl. 11, 12, 18, 22, 25 – a szálloda részére újrarendezed, a szobaszámok emeletenként egymáshoz képest továbbra is rendezettek maradnak: 11, 12, 22, 25, 18. Ennek az oka, hogy két szám sorrendje csak akkor változik, ha az új rendezési feltétel (a második számjegyre vonatkozó, amely a felvonótól való távolságot jelzi) szükséges.

Rendezési eljárások, melyeknél az elemek a korábbi sorrendet megőrzik, „stabil rendezési eljárásnak” nevezzük. Gyakorlatban ezek nagyon hasznosak, mint például az e-mail-ek esetében tudunk dátum, feladó vagy tárgy alapján rendezni. A leveleket először dátum, aztán tárgy szerint rendezzük, s így a tárgy szerinti rendezésnél egy tárgyon belül maradnak dátum szerint rendezve a levelek. Ez egyértelműnek tűnik, de csak stabil rendezés esetében működik.

Linkek:

[http://hu.wikipedia.org/wiki/Rendez%C3%A9s_\(programoz%C3%A1s\)](http://hu.wikipedia.org/wiki/Rendez%C3%A9s_(programoz%C3%A1s))



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Rossz csempe (2014-FI-04)

A SzámítógépKlub házának egyik szobáját ki akarják kövezni. 9-szer 9 fekete és fehér csempét akarnak lefektetni.

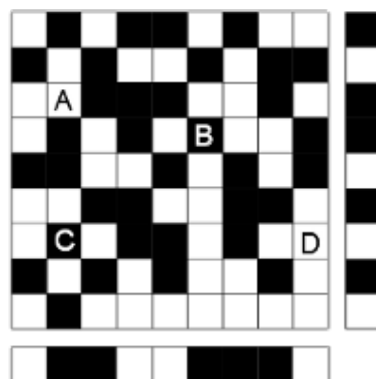
Egy belsőépítész tervezi meg a mintát. Jobbra és le egy-egy plusz csempesort tesz.

Ha egy sorban a fekete csempék száma páros, akkor a plusz mező jobbra ugyancsak fekete lesz. Egyébként fehér.

Ha egy oszlopban a fekete csempék száma páros, akkor a plusz mező lent ugyancsak fekete lesz. Különben fehér.

Sajnos hiba történt a csempézés során. A plusz mezők rendben vannak, de egy csempe rossz. Melyik?

- A. Az A csempének feketének kellene lennie.
- B. A B csempének fehérnek kellene lennie.
- C. A C csempének fehérnek kellene lennie.
- D. A D csempének feketének kellene lennie.



„C” válasz a helyes:

A C csempének fehérnek kellene lennie.

Felülről a hetedik sorban a fekete csempék páros számúak(4), de a plusz mező fehér. Tehát ebben a sorban kell lennie a rossz csempének. A többi sornál a plusz mezők helyesek.

Balról a második oszlopban páratlan számú (5) fekete csempe van, de a plusz csempe fekete. Tehát ebben az oszlopban kell lennie a rossz csempének. A többi oszlopban a plusz mezők helyesek.

A C-vel jelölt csempe van a hetedik sor negyedik oszlopában, tehát ez a rossz csempe.

Ez informatika!

Ez a feladat egyszerű példája a hibatűrő kódolásnak. A csempék reprezentálják a két lehetséges állapotukkal a biteket a kódban.

A kontrollmezőkkel (plusz sor és oszlop) minden sorban és oszlopban a fekete csempének páratlan számúnak kell lennie. Abból indulunk ki, hogy egyszerre csak egy bit mehet tönkre. Minden kódnál megvan a hibatűrés határa.

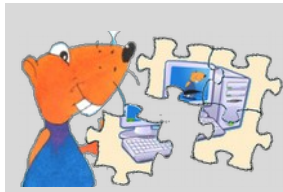
A sorok vagy oszlopok vizsgálata csak azt adja meg, hogy létezik egy hibás bit.

A sorok és oszlopok együttes vizsgálata határozza meg a hiba helyét és a javítását.

Az informatikában az információ tárolására és továbbítására sok kódot ismerünk különböző hibatűréssel. Néhány alkalmazásnak magasabb szintű adatbiztonságra van szüksége (pl. weben keresztül vásárlás), mint másoknak (pl. vidám macskás videók küldözgetése).

Linkek:

<http://hu.wikipedia.org/wiki/Hibajav%C3%ADt%C3%A1s>

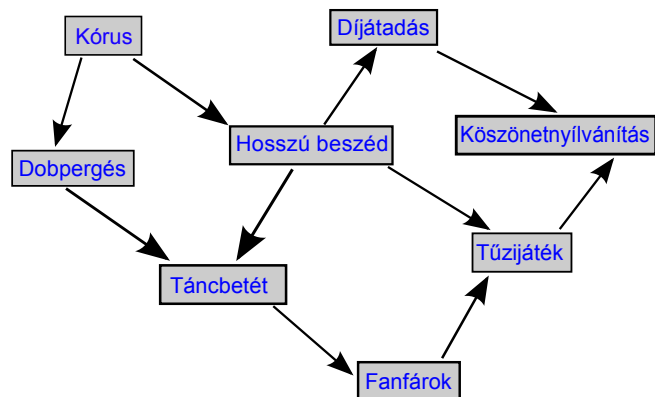


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Ünnepély (2014-FR-01)

Egy ünnepély egyedi eseményekből áll, melyeket helyes sorrendben kell végrehajtani. A képen az egyes események vannak. Egy nyíl egy eseményből a másikba azt jelenti, hogy az elsőnek a második előtt kell megtörténnie.

Például a kórusnak a dobpergés és a hosszú beszéd előtt kell énekelnie.



Mi az utolsó esemény az ünnepélyen?

- A. Tűzijáték
- B. Díjátadás
- C. Köszönetnyilvánítás
- D. Fanfárok

„C” válasz a helyes:

Egy ünnepély a következő előírás alapján szervezhető: amíg van esemény, amelybe nyíl mutat már lezajlott eseményekből, hajtsd végre. Ezen leírás alapján a kórus szereplése az első lehetséges esemény. Aztán következhet választhatóan a dobpergés vagy a hosszú beszéd. És ez így megy tovább, amíg az utolsó eseményre, a köszönetnyilvánításra nem kerül a sor.

Több lehetséges sorrend is kialakítható.

Ez informatika!

A teljesen hétköznapi helyzetekben is vannak sorrendiséget igénylő kapcsolatok tevékenységek között: felöltözésnél a zokni előbb kerül a lábra, mint a cipő és az alsó(nadrág) a nadrág alá kell kerüljön. A nadrágot sem húzzuk át a cipőn. De az, hogy először a zoknit vagy az alsó(nadrágo)t húzzuk fel, mindegy.

Ha a felöltözés sorrendjében minden rendben volt, akkor azt mondjuk, hogy az események sorrendje topológikusan rendezett.

Az informatikában a topológikus rendezés fontos: például az olyan programrészeket, melyek végrehajtásához egy másik programrész eredménye szükséges, úgy kell végrehajtanunk, hogy a szükséges adatok mindig rendelkezésre álljanak.

Egy további példa: egy adatbankból egy bejegyzés törléséhez előbb minden rá hivatkozó bejegyzést törölni kell.

Ha egy topológikus rendezést találunk, akkor biztosítva kell legyen, hogy nincs kölcsönös sorrendiségi kapcsolat. Az ilyen „ciklusok” az egész folyamatot blokkolhatják.

Linkek:

http://hu.wikipedia.org/wiki/Topologikus_sorrend

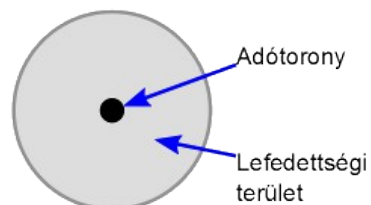


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Viharbiztos hálózat (2014-HU-02)

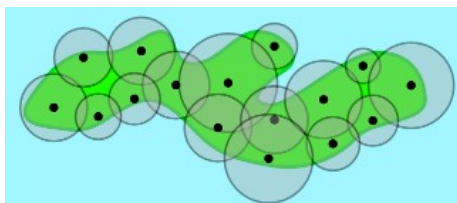
Egy viharzónával körülvett szigetre mobiltelefon-tornyokat akarnak telepíteni. Minden torony egy kör alakú területet fed le.

Ha két torony lefedési területe fedi egymást, akkor a tornyok közvetlen rádióhullám-kapcsolatban vannak. Egy adótorony közvetetten is kapcsolatban lehet egy másikkal: közvetlenül egymással összekötött tornyok láncán keresztül.

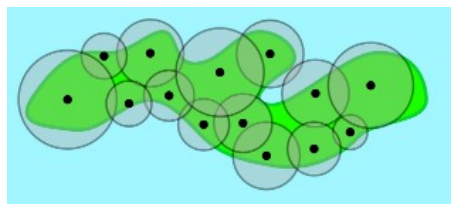


Az állandó viharok miatt a tornyokat úgy akarják felállítani, hogy egy torony kiesése a lehető legkisebb kárt okozza. Ha kiesik egy torony, akkor az összes többi ennek ellenére összeköttetésben maradjon.

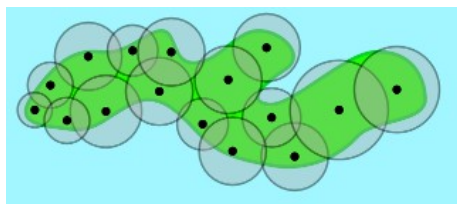
Hogy állítsák fel a tornyokat?



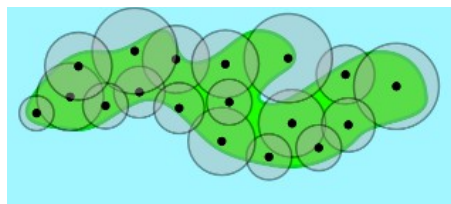
A.



B.



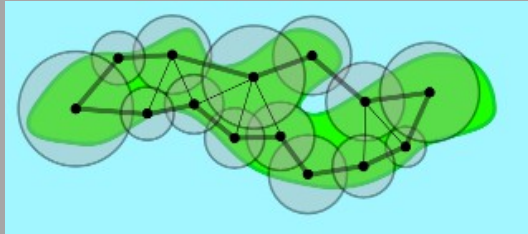
C.



D.

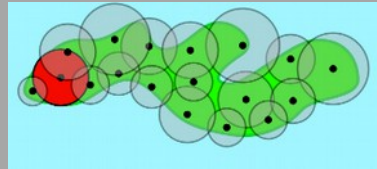
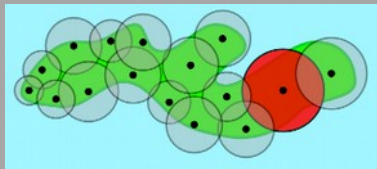
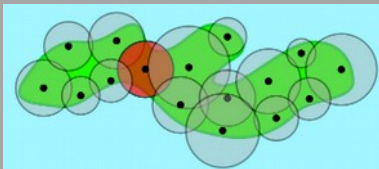
„B” válasz a helyes:

Ha felrajzoljuk a közvetlen kapcsolatokat a térképre, akkor többek közt egy gyűrűformájú kapcsolat is létrejön (ld. vastag vonal):



Ha eltávolítunk egy tetszőleges tornyot, akkor a többi ennek ellenére összeköttetésben marad.

A többi térképen ilyen kapcsolat-gyűrű nem található, de lesz egy kritikus torony (piros), amelyik ha kiesik, akkor két csoportra bontja a tornyokat.

**Ez informatika!**

A műszaki rendszereknél kieshetnek bizonyos összetevők. A mobiltelefon-hálózat egy ilyen rendszer, melynek többek között az adótornyok az összetevői.

Amennyiben kiesik egy összetevő, az különböző súlyos következményekkel járhat. Hirtelen az egész rendszer használhatatlanná válhat, vagy csak bizonyos funkciók lesznek működésképtelenek, míg a többi tovább működik. Vagy minden működik tovább.

A kiesés biztonságának foka attól függ, hogy a rendszer összetevői milyen erősen kapcsolódnak egymáshoz és milyen különleges pótalkatrészek (összetevők) kerülnek beépítésre, melyek a kiesésekkor bekapcsolódnak a működésbe.

Az informatika mindenekelőtt a műszaki rendszerek szoftver-összetevőieért felelős. Ha egy szoftver-összetevő programozási hibát tartalmaz, vagy elfogadhatatlan bemenetekkel „traktálják”, rossz eredményeket adhat ki, vagy teljesen össze is omolhat.

Ekkor nem használ, ha a műszaki rendszerben egy ugyanolyan programhibát tartalmazó pót-számítógép kerül beállításra. Ahhoz, hogy egy műszaki rendszer különösen biztosan működjön, a pótprogramot egy másik csapatnak kell másodszor is megterveznie és egy másik programnyelven megvalósítania (sokféleségi redundancia).

Linkek:

http://hu.wikipedia.org/wiki/H%C3%A1l%C3%B3zati_topol%C3%B3gia

http://en.wikipedia.org/wiki/Single_point_of_failure

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

	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Felfele a folyón (2014-JP-01)

Hogy elérje a célját a hódnak a folyórendszer megfelelő útján kell felúsznia. Az útján akadályokat kell leküzdenie. Ehhez a következő energiákat használja fel:

Akadály	szükséges energia
	2 ág
	3 ág
	5 ág

Hogy elég energiája legyen, induláskor a hód 15 ágot eszik.

A képen lévő folyórendszeren láthatod az akadályokat. A, B, C, D és E a lehetséges utak közötti állomások.



Az alábbi utak közül melyiken menjen a hód? Figyelj arra, hogy csak 15 ágnyi energiája van.

- A. Start → A → C → E → Cél
- B. Start → A → C → E → D → Cél
- C. Start → B → C → D → E → Cél
- D. Start → B → C → D → Cél

„C” válasz a helyes:

A különböző utak a következő energiaszükségleteket jelentik:

A) Start → A → C → E → Cél: $2+5+5+5=17$

B) Start → A → C → E → D → Cél: $2+5+5+2+3+5=22$

C) Start → B → C → D → E → Cél: $3+3+2+2+5=15$, az egyetlen út, amelyikhez nincs szükség több energiára, mint ami a hódnak van.

D) Start → B → C → D → Cél: $3+3+2+3+5=16$

Ez informatika!

A folyórendszer egy hálózat, ahol a köztes állomások (A-tól E-ig) és a Start, valamint a Cél az úgynevezett csomópontok. Az akadályok energiaszükséglete felfogható úgy, mint a távolság két összekötött csomópont között. Így a hódnak a Starttól a Célig vezető legrövidebb utat kell megkeresnie.

A legrövidebb út keresésének problémájára a legismertebb a Dijkstra algoritmus, de ismert még a Floyd és a Warshall algoritmus is, mely minden egyes csomópontból az összes többi csomópontba a legrövidebb út hosszát állapítja meg. A navigációs rendszereknél már találkozhattál is ezeknek az algoritmusoknak az alkalmazásával.

Linkek:

<http://hu.wikipedia.org/wiki/Dijkstra-algoritmus>

http://hu.wikipedia.org/wiki/Edsger_Wybe_Dijkstra

	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Melyik fénykép (2014-JP-03)

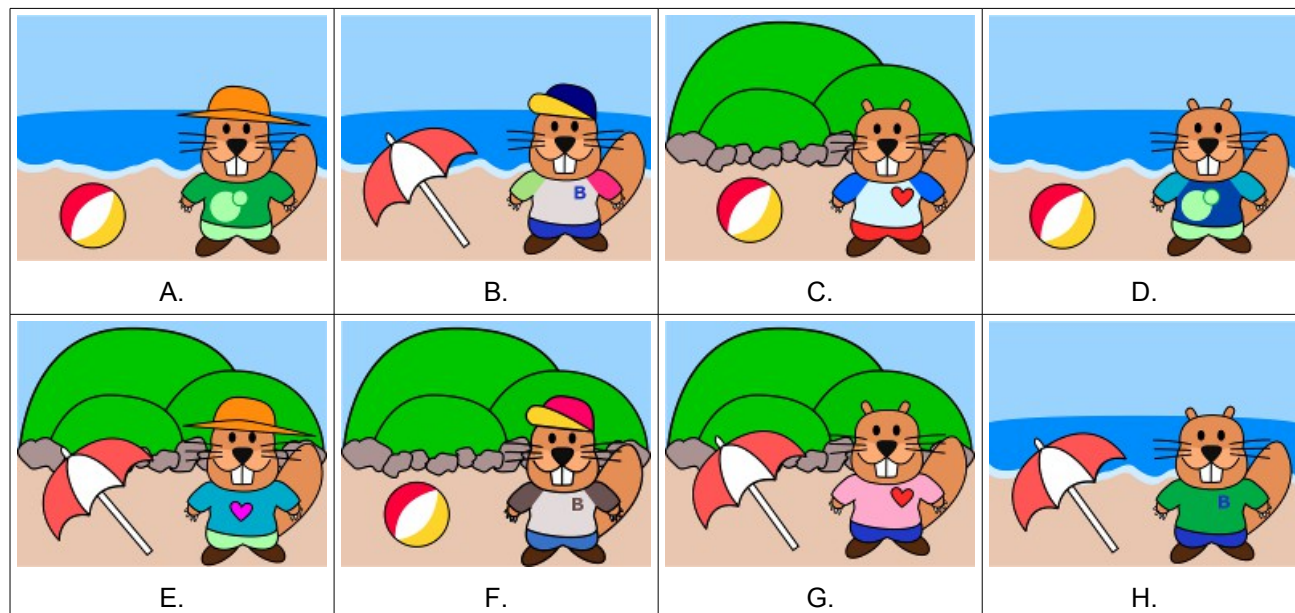
János 8 fényképet készített. Ezek közül egyet Bellának szeretne adni. Ki akarja találni, hogy Bella melyik fotót szeretné.

Ehhez kérdéseket tesz fel:

„Napernyővel készült fényképet szeretnél?” – „Igen.”

„Olyan fényképet szeretnél, amin sapkát vagy kalapot hordok?” – „Nem.”

„Olyan fényképet szeretnél, amin látszik a tenger?” – „Igen.”

Melyik fényképet szeretné Bella?

„H” válasz a helyes:

A B, E, G és H képek felelnek meg Bella János első kérdésére adott válaszára.

A C, D, G és H képek felelnek meg Bella második kérdésre adott válaszára.

Az A, B, D és H képek felelnek meg a harmadik kérdésre adott válaszára.

Csak a H kép felel meg mindhárom kérdésnél.

Ez informatika!

Adatok tárolásához és feldolgozásához a számítógépek biteket használnak, melyek csak két különböző állapotot vehetnek fel: „be” vagy „ki” (azaz „igaz” vagy „hamis”; „igen” vagy „nem”; 1 vagy 0).

Ebben a feladatban Bella kívánságát 3 bittel írhatjuk le: minden egyes kérdéshez, amit János feltett egy bit. Bella válasza azt jelenti, hogy az első bit „be” ÉS a második „ki” (azaz „NEM be”) ÉS a harmadik „be” van kapcsolva.

Az informatikában a logikai műveletek, mint az ÉS és a NEM elegendőek ahhoz, hogy egy bit értékét minden tetszőleges módon egy másik bit-értékre módosítsuk.

Mindent, amit a számítógép elvégez, ezekkel az egyszerű műveletekkel elérhetünk – például dolgok azonosítása (itt egy fénykép) egy adatgyűjteményből (János 8 fényképe).



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

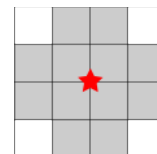
Rádióhálózat a faluban (2014-JP-06)

Egy faluban rádióhálózatot telepítenek több adótoronnyal. Ez szolgáltatja a lakosoknak az internetet.

Minden adóállomásnak egy meghatározott sugárzási és vételi területe van.

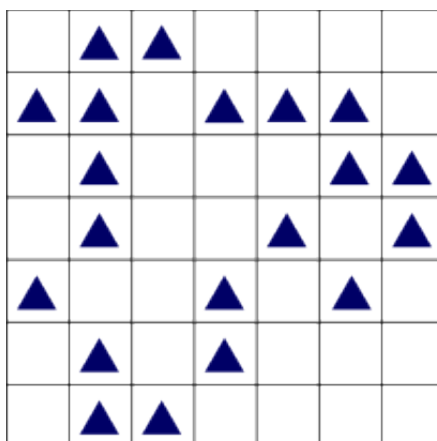
Minden állomás egy meghatározott területen belül tud jelet sugározni és fogni.

Ahogy a kép is mutatja, a középső toronyból (piros csillag) csak 12 környező telken (szürke) fogható a jel.



Egy adótorony mindig csak két telek határának kereszteződésébe állítható. A tornyok adás és vételi területei egymást fedhetik.

A kép a falu térképét mutatja. Minden háromszög egy házat ábrázol.



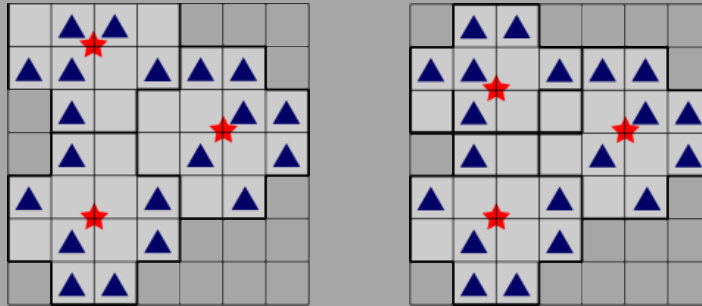
Legalább hány adótoronyot kell telepíteni ahhoz, hogy minden házban legyen kapcsolat a hálózattal?

- A. 2 tornyot.
- B. 3 tornyot.
- C. 4 tornyot.
- D. 5 tornyot.

„B” válasz a helyes:

Két adótoronnyal nem fedhetünk le minden házat.

Három adótoronnyal két különböző módon is megvalósíthatjuk, hogy az összes ház kapcsolódhasson a hálózathoz.

**Ez informatika!**

Az informatika ismer olyan algoritmikus eljárásokat, melyekkel nagy összefüggő területek kis, különböző formájú területekkel költséghatékonyan és többé-kevésbé lefedhetők.

Például a ruhaiparban az anyagok kiszabása vagy a lemez-alkatrészek lyukasztása a gépiparban. A lehetséges alkalmazásokhoz tartozik a mobiltelefonok, a digitális televíziós és rádiós műsorszórás vagy a WLAN lefedettségének tervezése is.

Ezek az eljárások, melyek ilyen alkalmazásokban minden megadott adatra működnek és mindig a legjobb megoldást biztosítják, gyakran használtak. Ez azt jelenti, hogy nagyobb adathalmazra tovább tart a megoldás kiszámítása, mint azoknak az élete, akik várják a megoldást.

Az informatikában ilyenkor olyan eljárásokkal dolgoznak, melyek nem mindig a legjobbak, de megbízhatóan jó megoldást találnak – és ezért hatékonyabbak, különösen mivel gyorsabbak.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Perecek (2014-LT-07)

Két Hód dolgozik egy pékségben. Zsuzsa, a péklány egyszerre mindig három perecet vesz ki a kemencéből és ráakasztja jobbról egy botra: először egy A, aztán egy B végül egy O alakú perecet.

Péter, az eladó mindig a jobb oldalon lógó perecet adja el. Zsuzsa gyorsabban süti a pereceket, mint ahogy Péter eladja azokat.

Legalább (legkevesebb) hány perecet adott el Péter, hogy a bot a képnek megfelelően nézzen ki?



Melyik gyerek lesz a legfelső mezőn?

- A. 5 perecet.
- B. 7 perecet.
- C. 9 perecet.
- D. 11 perecet.

„C” válasz a helyes:

Zsuzsának legalább hatszor 3 peracet (18db) kellett a rúdra akasztania, mivel 6db A alakú perac maradt a boton. Összesen azonban 9 perac van még a boton, így Péter legalább 9 darabot adott el: 4 B alakút és 5 O alakút.

Hogy hány ABO-perac hármast adott még el Péter, az ismeretlen marad.

Ez informatika!

A bot egy úgynevezett Verem (stack), mely az informatikában egy tárolási elgondolás. Itt az új információ csak a „legfelső” információra kerülhet rá (push) és mindig csak a „legfelső” információ vehető ki (pop).

A boton az új perecek csak jobbra az első helyre kerülhetnek és onnan is emelhetők le. Itt a „legfelső” a verem-elgondolásban mint „jobbról a legelső” került megvalósításra.

A verem tárolásnál a hozzáférést LIFO-nak is nevezzük (Last In First Out, azaz utolsónak be elsőnek ki)

Linkek:

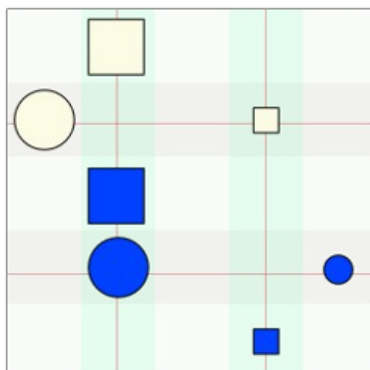
[http://hu.wikipedia.org/wiki/Verem_\(adatszerkezet\)](http://hu.wikipedia.org/wiki/Verem_(adatszerkezet))



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Igaz vagy hamis (2014-RU-02)

Alíz és Tomi „igaz vagy hamis” játékot játszanak az osztályterem mágnes táblájánál. Alíz hét különböző mágneset tesz a táblára.



Ezután állításokat mond a mágnesek alakjáról, színéről, nagyságáról és elhelyezkedéséről.

Egy állítás igaz, a többi hamis. Tamásnak ki kell találnia, melyik állítás igaz.

Melyik állítás igaz?

- A. Van két mágnes X és Y úgy, hogy X sötétkék és Y világossárga és X Y felett van.
- B. Minden tetszőleges két X és Y mágnesre igaz, hogy ha X egy négyzet és Y egy kör, akkor X Y felett van.
- C. Minden tetszőleges két X és Y mágnesre igaz, hogy ha X kicsi és Y nagy, akkor X jobbra van Y-től.
- D. Minden tetszőleges két X és Y mágnesre igaz, hogy ha X világossárga és Y sötétkék, akkor X Y alatt van.

„C” válasz a helyes:

mivel a kis mágnesek mind jobbra vannak a nagy mágnesektől.

Az A válasz hamis, mert nincs olyan sötétkék mágnes, ami világossárga mágnes felett lenne.

B válasz hamis, mivel nem minden négyzet alakú mágnes található kör alakú mágnesek felett.

D válasz hamis, mivel nem minden világossárga mágnes található sötétkék mágnes alatt.

Ez informatika!

Ebben a HÓD feladatban arról van szó, hogy megállapítsuk, egy állítás igaz vagy hamis.

Egy mágnes tulajdonságait „négyzet alakú(X)”, „kör alakú(X)”, „nagy(X)”, „kicsi(X)”, „sötétkék(X)”, „világossárga(X)” predikátumokkal írhatjuk le.

Két mágnes közötti kapcsolatot pedig a „felette(X,Y)”, „alatta(X,Y)” és „jobbra(X,Y)” predikátumokkal írhatjuk le.

A predikátum (elsőrendű) logika formális nyelvén a kijelentések így néznek ki:

A) létezik X, Y: sötétkék(X) és világossárga(Y) és felette(X,Y)

B) minden X, Y-ra: (négyzet alakú(X) és kör alakú(Y))-ből következik, hogy felette(X,Y)

C) minden X, Y-ra: (kicsi(X) és nagy(Y))-ből következik, hogy jobbra(X,Y)

D) minden X, Y-ra: (világossárga(X) és sötétkék(Y))-ből következik, hogy alatta(X,Y)

Az informatikában vannak olyan programnyelvek, melyekben az elsőrendű (predikátum) logikai állításokkal programozunk. Például a Prolog egy ilyen logika-vezérelt programnyelv.

Linkek:

<http://hu.wikipedia.org/wiki/Prolog>



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

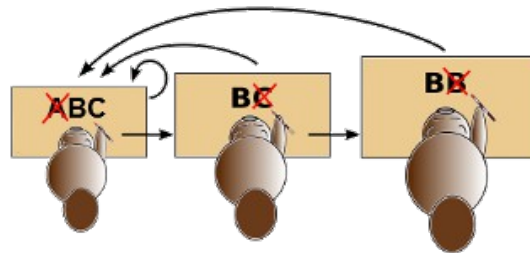
Hírek Hódiából (2014-RU-05)

A távoli Hódiában az újságcikkek – melyek különben is csak az A, B és C betűkből állnak – a Hírhivatal által „módosításra” kerülnek. A Hivatal három korrektora elolvas egy-egy hírt jobbról balra és meghatározott betűmintákat keresnek benne.

1. Az Alkorrektor az ABC sorozatokat keresi. Ha talál egyet, akkor BC-re cseréli és a hír olvasását újra előről kezdi. Ha már nem talál ilyen sorozatot, akkor átadja a szöveget a Főkorrektornak.

2. A Főkorrektor a BC sorozatokat keresi. Ha talál egyet, akkor B-re cseréli és visszaadja a módosított szöveget az Alkorrektornak. Ha már nem talál ilyen sorozatot, akkor átadja a szöveget a Legfőbb korrektornak.

3. A Legfőbb korrektor a BB sorozatokat keresi. Ha talál egyet, akkor B-re cseréli és visszaadja a módosított szöveget az Alkorrektornak. Ha már nem talál ilyen sorozatot, akkor vége a korrektúrázásnak.



A következő üzenetek közül három a korrektúrázás végén csak egy B betűből áll. Melyik NEM?

- A. AAABCB
- B. ABCABC
- C. ABABCB
- D. ABCCCC

„C” válasz a helyes:

Az egyes hírek a következőképpen kerülnek korrektúrázásra:

- A) AAABCB → AABCB → ABCB → BCB → BB → B
- B) ABCABC → BCABC → BCBC → BBC → BB → B
- C) ABABCB → ABBCB → ABBCB → ABB → AB
- D) ABCCCC → BCCCC → BCCC → BCC → BC → B

Ez informatika!

Mit lehet kiszámítani? Ezt a kérdést – különösen a 20. század elején – sok tudós feltette magának. Többen figyelembe vették annak a lehetőségét, hogy a számítások módját, illetve a számítási módszerek/eljárások (azaz az algoritmus) fogalmát egy formális modellel írják le.

A legismertebb ilyen modell a Turing-gép, mely a nevével ellentétben soha nem került megépítésre. Nem ennyire ismertek a szöveg cserélő rendszerek, amit az orosz Andrej Markov írt le. Egy ilyen szövegcsereológ rendszert használtak a hódiai korrektorok.

A szép és megnyugtató az informatikában, hogy a számítási eljárások (módszerek) összes eddig fejlesztett formalizálását (pl. továbbiak a Lambda-kalkulus, vagy a μ -rekurzív függvények) egyenértékűnek ismerték el.

Alapvetően a modern számítógépek sem többet, sem kevesebbet nem tudnak, mint a formális modellek. Ennyit az elméletről; a modern programnyelvek és fejlesztési környezetek létezése ezután egy következő lépés a gyakorlatban.

Linkek:

[http://hu.wikipedia.org/wiki/Andrej_Andrejevics_Markov_\(matematikus,_1856%E2%80%931922\)](http://hu.wikipedia.org/wiki/Andrej_Andrejevics_Markov_(matematikus,_1856%E2%80%931922))

http://en.wikipedia.org/wiki/Markov_algorithm

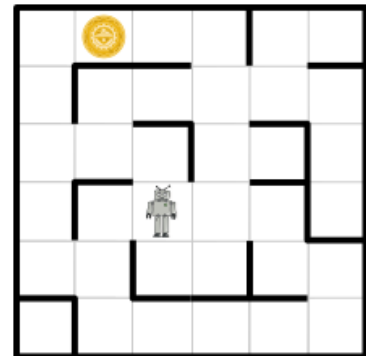


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Világűr labirintus (2014-SI-02)

Űrhajósok egy elhagyatott bolygón szállnak le. A tele-szemüvegükön rejtélyes képeket látnak. Követik a jelzéseket és egy robotot találnak. A robot egy labirintus közepén áll, amit az űrhajósok fentről jól megfigyelhetnek, és egyértelmű közelképeket sugároz a környezetéről.

A labirintus négyzetekre osztható és az egyik ilyen négyzetben található a robot. Egy másik négyzetben egy ismeretlen tárgy van. Az űrhajósok szeretnék a robotot ehhez a tárgyhoz irányítani és az általa sugárzott közelképeket megnézni.



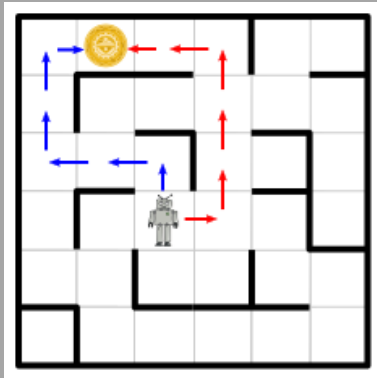
Hirtelen felvillan négy rejtélyes szövegsor összesen négy különböző szóval a tele-szemüvegen. A robot és a tárgy is felismerhető. Némi töprengés után az űrhajósok megállapítják: a szavak parancsok, melyek a robotot egy szomszédos négyzetbe irányítják; mind a négy lehetséges iránynak külön parancsa van. Ezenkívül az űrhajósok abban is biztosak, hogy az egyik szövegsor egy parancssor, amellyel a robotot a tárgyhoz lehet irányítani

A négy szövegsor közül melyik irányítja a robotot az ismeretlen tárgyhoz?

- A. Ha' poS poS Ha' Ha' nIH
- B. Ha' Ha' poS Ha'
- C. Ha' poS poS Ha' nIH Ha'
- D. Ha' poS nIH vl'ogh Ha' poS

„A” válasz a helyes:

Az utasítássorok egyike sem tartalmaz hatnál több utasítást. Minden utasítással a robot egy lépést tehet egy szomszédos négyzetbe. A kép azt a két lehetséges utat mutatja, amelyikben a robot hat lépésben el tud jutni a tárgyhoz:



A parancssornak tehát a robotot vagy a piros nyílon kell irányítania:

jobbra, fel, fel, fel, balra, balra.

Ehhez azonban egyik utasítássor sem megfelelő.

A másik a kék nyíllal jelzett út: fel, balra, balra, fel, fel, jobbra. Ehhez az A parancssor pont illeszthető.

A B lehetőséget eleve kizárhatjuk, mivel 4 lépésben nem érhető el a tárgy a robot helyéről.

Ez informatika!

A kriptográfia a titkosított üzenetek elolvasásának tudománya. Már a görögök is igyekeztek titkosítani az üzeneteiket. Ehhez a titkosított üzenet lehetséges jelentéseinek ismerete is hozzátartozott.

A második világháborúban az Enigma-gép által titkosított üzeneteket úgy próbálták megfejteni, hogy az üzenetekben német városneveket, időjárásjelentésekben előforduló szavakat kerestek. Mivel a fontos üzenetek legtöbbször időjárásjelentéssel kezdődtek.

Ebben a feladatban, mint „kriptoanalitikus”, azaz üzenettitkosító kellett tevékenykedned. A megfejtés tudományosan könnyebb, mint klingoni nyelven beszélni ;)

Linkek:

<https://hu.wikipedia.org/wiki/Kriptogr%C3%A1fia>

[https://hu.wikipedia.org/wiki/Enigma_\(g%C3%A9p\)](https://hu.wikipedia.org/wiki/Enigma_(g%C3%A9p))

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



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Öntözés (2014-SI-07)

Amikor a szelep zárva van, a víz nem tud átfolyni.

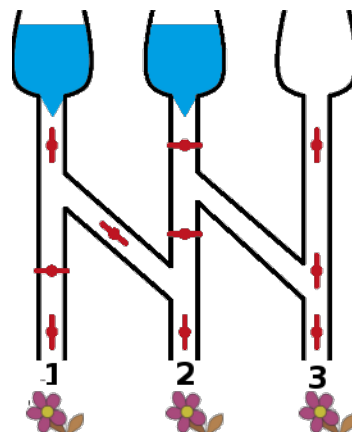


Amikor a szelep nyitva van, a víz átfolyik.



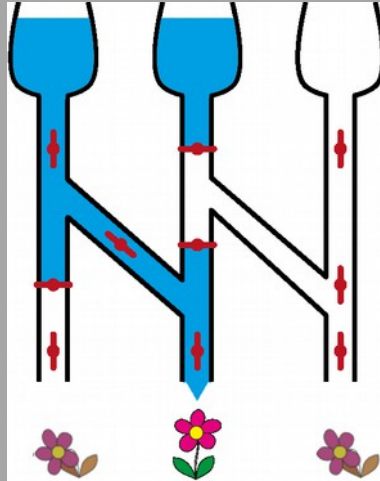
A három szomszjas virág közül melyik kap vizet, ha a szelepek így állnak?

- A. 1. virág.
- B. 2. virág.
- C. 3. virág.
- D. Egyik sem.



„B” válasz a helyes:

Csak a középső virág kap vizet. Lásd az ábrát.

**Ez informatika!**

Informatikában ez az öntözőrendszer egy kapcsolás. A szelepek a kapcsolók, amelyeknek két állásuk van: be vagy ki vannak kapcsolva. A bemeneti tölcsek és a kapcsolóállások szerint terjed az információ (mint ahogy folyik vagy nem folyik a víz) a kapcsolókon keresztül a virágokig.

Az elektronikus készülékek elektronikus kapcsolásokat tartalmaznak, melyen keresztül az elektromosság folyik. Az üvegszálás kapcsolásokon keresztül az információ mint lézerfény közlekedik.

Egy robot eszköznek, melynek a környezetben kell dolgoznia, az alkatrészei, kapcsolói hamar tönkre tudnak menni: erős mágneses hatás, sok nedvesség, szélsőséges hőmérséklet. Az ilyen robotoknak robusztus kapcsolókra van szüksége, melyekben hidraulikus olaj vagy sűrített levegő van.

Az automatizálási technikában és a robotikában azonban a processzor-perifériákkal ellentétben (szenzorok és aktorok) folyadékos és hidraulikus kapcsolások vannak érvényben jelenleg is.



benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Fogkefék (2014-SI-08)

„Ne olyan gyorsan” - mondja a hódmama.

„Éva és Csaba, cseréljétek fogkefét! Anna és Csaba, ti ketten szintén!” De ezután már nem tudja, hogyan tovább.



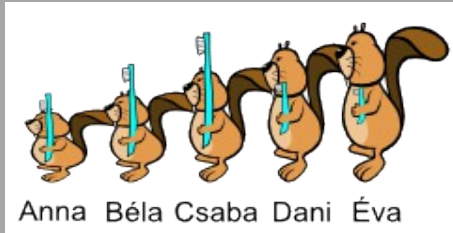
Anna Béla Csaba Dani Éva

Melyik két hódnak kell még a fogkefáját elcserélnie, hogy mindenkinek a méretéhez való fogkeféje legyen?

- A. Béla és Csaba
- B. Béla és Dani
- C. Anna és Éva
- D. Senkinek sem kell cserélnie.

„B” válasz a helyes:

Az első felállás:



„Éva és Csaba cseréljétek fogkefét!”



„Anna és Csaba, ti ketten szintén”



Most már csak Bélának és Daninak kell cserélnie

Ez informatika!

A programozók gyakran olyanok, mint hódmama, aki ügyel a sorrendre.

De fogkefék helyett számokat mozgatnak a számítógép memóiahelyei között. Adatok cseréje a programozás alpműveletei közé tartozik.

Sokszor kell egy számsort nagyság szerint rendezni. A számokat egymást követő cellákban tárolják. A számítógépes programnak arról kell gondoskodnia, hogy a legkisebb szám az első cellába kerüljön, a második legkisebb a másodikba, és legvégül a legnagyobb szám az utolsó cellába.

Ez a rendezés megvalósítható, ha többször cseréljük a tárolócellák tartalmát.



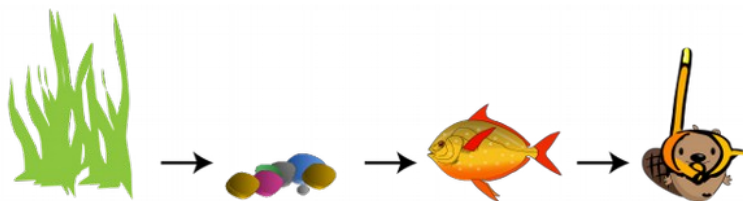
benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Ragasztott rajzok (2014-SK-01)

János rajzolt egy akváriumot.

Feldíszítette kis ragasztott rajzokkal.

Először ragasztott hínárt, majd köveket. Ezután a halat és végül a búvárhódot.



Hogy nézett ki a kép?



A.



B.



C.



D.

„A” válasz a helyes: a képek itt vannak egymáson a helyes sorrendben.

B rossz, mert a bűvárhód nincs egészen elől, rajta van a hal.

C rossz, mert a hínár nincs egészen hátul. A hal mögötte van.

A D válasz rossz, nem a hínár előtt úszik, hanem közöttük.

Ez informatika!

Az informatika több területén fontos szerepe van a sorrendnek. Itt például egy kép esetében, ami több képelemből tevődik össze.

Egy másik sorrend más képet eredményezne, habár az egyes képelemek ugyanazok maradnának.

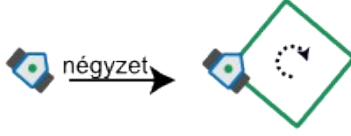
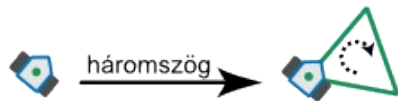
	benjamin	nehéz	közepes	könnyű
	kadét	nehéz	közepes	könnyű
	junior	nehéz	közepes	könnyű
	senior	nehéz	közepes	könnyű

Rajzbot (2014-SK-07)

A Rajzbot robot tud haladni és közben rajzolni.

A következő parancsokat adhatjuk Rajzbotnak: négyzet, háromszög, előre, fordulj

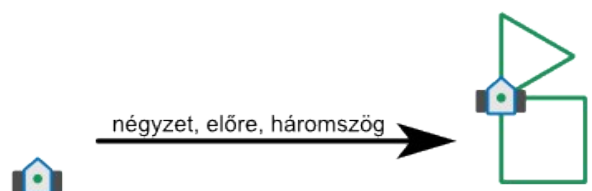
Az egyes parancsok hatása a következő:

négyzet	Rajzbot egy négyzetet rajzol, a csúcsainál jobbra fordulva.	
háromszög	Rajzbot egy háromszöget rajzol. A csúcsoknál jobbra fordul.	
előre	Rajzbot előre megy egy már megrajzolt vonalon a következő csúcsig.	
fordulj	Rajzbot jobbra fordul a következő megrajzolt vonalig.	

Rajzbotnak több parancsot is kiadhatunk egymás után.

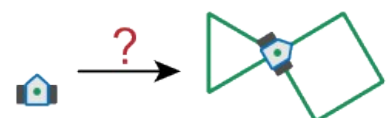
Például: négyzet, előre, háromszög.

A parancsok hatását a jobboldali ábrán láthatod.

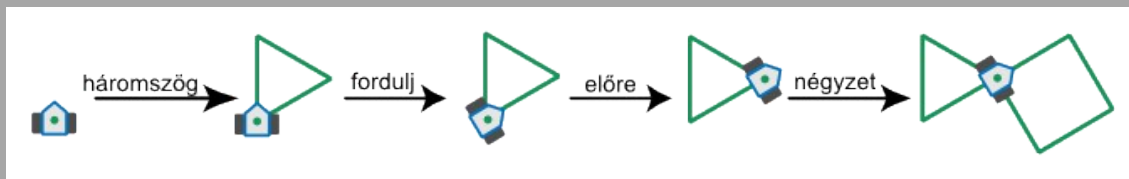


Melyik parancssorral rajzoltathatjuk meg Rajzbottal a következő ábrát?

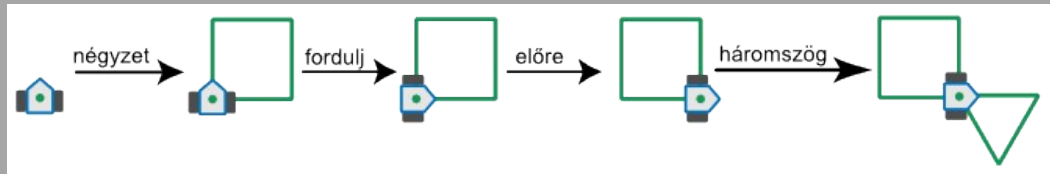
- A. négyzet, fordulj, előre, háromszög
- B. háromszög, fordulj, előre, négyzet
- C. háromszög, fordulj, négyzet
- D. négyzet, előre, négyzet, fordulj, háromszög



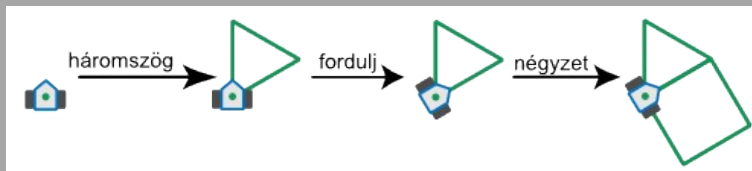
„B” válasz a helyes:



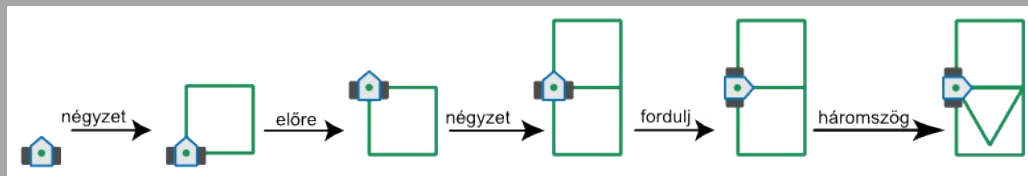
Az A válaszban a háromszög és a négyzet parancsokat felcseréltük:



A C válaszban hiányzik az előre:



A D válasz nyilvánvalóan rossz, hiszen két négyzetet rajzoltattunk:



Ez informatika!

A robotok (és a számítógép) programozásának legegyszerűbb építőkövei a parancsok és a parancssorok. Mivel a valódi robotok általában nem rajzolnak, hanem autókat szerelnek össze vagy az orvosi gyógyászatban segítenek, sokkal több és érthetően bonyolultabb parancsot ismernek, mint Rajzbot. A parancsaik hatása is sokkal erősebb, ezért is fontos, hogy a programozóik nagyon pontosan dolgozzanak.

De Rajzbot egyszerű rajzutasításaival jól meg lehet tanulni programozni. Ehhez hasonló parancsokat legelőször az amerikai informatikus Seymour Papert vezetett be a Logo programozási nyelvbe. Logo-ban egy kis teknőc (turtle) rajzol. Innen alakult ki a teknőc-grafika, amit több programozási nyelv is használ (mint pl. a Python).

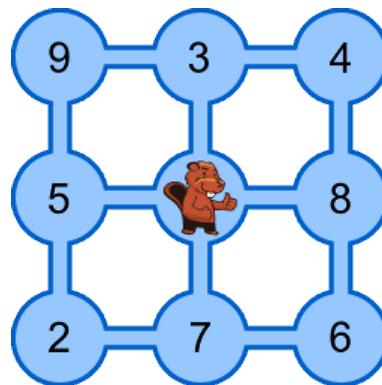


benjamin	nehéz	közepes	könnyű
kadét	nehéz	közepes	könnyű
junior	nehéz	közepes	könnyű
senior	nehéz	közepes	könnyű

Sok barát (2014-UA-04)

A képen 9 tavat látsz, melyek csatornákkal vannak összekötve. Hód Tomi a középső tóban él. Barátai meg a szomszédos tavakban. A számok a képen azt mutatják, hogy Tominak hány barátja él az egyes tavakban.

Tomi szeretné meglátogatni a barátait. Otthonról indul és minden nap átúszik egy csatornán egy másik tóba, ahol meglátogatja a barátait és ott is alszik. A következő napon továbbúszik.



Maximum hány különböző barátját tudja Tomi meglátogatni 4 nap alatt?

Mindegy, hogy a negyedik napon melyik tóba kerül!

- A. 21 barátját.
- B. 24 barátját.
- C. 25 barátját.
- D. 30 barátját.

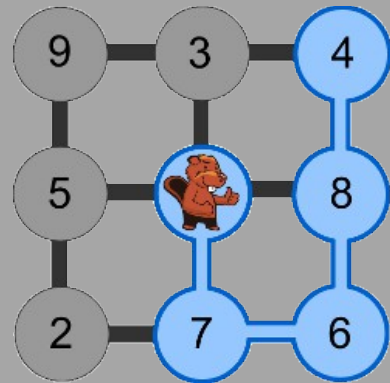
„C” válasz a helyes:

Tomi 4 nap alatt 25 barátját tudja meglátogatni. Az útját (7-6-8-4) a mellékelt ábrán kékkel jelöltük.

Az A és a B válaszok olyan utakból adódhatnak, ahol ugyan Tomi olyan tavakat látogat meg, ahol sok barátja él, de összesen mégis kevesebb baráttal találkozik.

A D válasz akkor lehetséges, ha a négy legnagyobb számot összeadjuk, de ezeket a tavakat 4 nap alatt Tomi nem tudja bejárni.

A tavakban élő barátok számának összes többi kombinációja, ahol összesen 25 barátnál több él, 4 napon belül nem érhető el.

**Ez informatika!**

Informatikában a tavak és csatornák által leírt rendszert gráfnak nevezzük. A tavak a gráf csomópontjai, a barátokat jelölő számok a csomópontok értékei, a csatornák a gráf élei.

Egy olyan összefüggő utat keresünk a gráfban, mely a következő feltételeket teljesíti:

1. Tomi tava a kezdő csomópont. Ennek értéke 0.
2. Az út maximum 4 élből állhat.
3. A csomópontok értékei, melyek az úton fekszenek a lehető legnagyobb legyen. Minden csomópont csak egyszer számít.

Ez például a fuvarvállalatoknak valós probléma. A vezető csak megadott számú órát vezethet naponta és csak megadott napon keresztül. Ezért az útját gondosan meg kell tervezni, hogy a lehető legtöbb árut szállíthassa ki és a jármű minél jobban kihasználásra kerüljön.

Viszonylag kis gráfokban, mint a feladatban is, a keresett út szisztematikus próbálgatással is megtalálható.

Nagyobb gráfoknál, melyek például 100 várost, mint csomópontot tartalmaznak, használhatunk kidolgozott eljárásokat, melyek jó megoldást találnak anélkül, hogy minden lehetséges változatot végig kellene próbálnunk.

	senior	junior	kadét	benjamin
könnyű	2014-CA-04	2012-DE-10	2014-CA-05	2012-FR-10
	2014-CH-01	2014-ES-03	2014-CH-05	2014-CZ-06
	2014-FI-04	2014-FI-02	2014-JP-01	2014-CZ-08
	2014-FR-01	2014-FI-04	2014-JP-03	2014-CZ-10
	2014-LT-07	2014-JP-06	2014-JP-06	2014-SI-07
	2014-SI-02	2014-SI-02	2014-SI-08	2014-SK-01
közepes	2012-FR-04	2014-CA-04	2014-DE-05	2014-CH-05
	2014-CA-07	2014-CA-07	2014-DE-08	2014-ES-03
	2014-CH-02	2014-CH-01	2014-ES-02	2014-JP-01
	2014-DE-02	2014-DE-08	2014-FI-02	2014-JP-03
	2014-DE-07	2014-FR-01	2014-SI-02	2014-JP-06
	2014-HU-02	2014-LT-07	2014-SK-07	2014-SI-08
nehéz	2011-DE-14	2012-FR-04	2014-CA-04	2014-CA-05
	2014-AT-03	2014-CH-02	2014-CA-07	2014-DE-05
	2014-AU-01	2014-DE-02	2014-CH-01	2014-DE-08
	2014-CH-07	2014-DE-07	2014-FR-01	2014-ES-02
	2014-RU-03	2014-HU-02	2014-HU-02	2014-SK-07
	2014-RU-05	2014-RU-03	2014-LT-07	2014-UA-04

Támogatóink:

