Az autodidakta/kritikus tanulás mintázatai – II. rész – továbbra is a just-in-time-tudásmenedzsmentről

(Patterns of the autodidactic/critic-oriented learning – Part II)

Pitlik László, KJE/MY-X team

Kivonat: A sorozat második részében az alapjegyzet súlypontjai mellé új súlypontok, nézőpontok kerülnek bemutatásra annak érdekében, hogy a BPROF által elvárt gyakorlatiasság, egy fajta just-in-time-tudásmenedzsment egyre jobban közelíthető legyen.

Kulcsszavak: tudásmenedzsment, oktatásdidaktika, gyakorlat-orientáltság, duális képzés, tantervkészítés, akkreditáció, optimalizálás, minőségmenedzsment

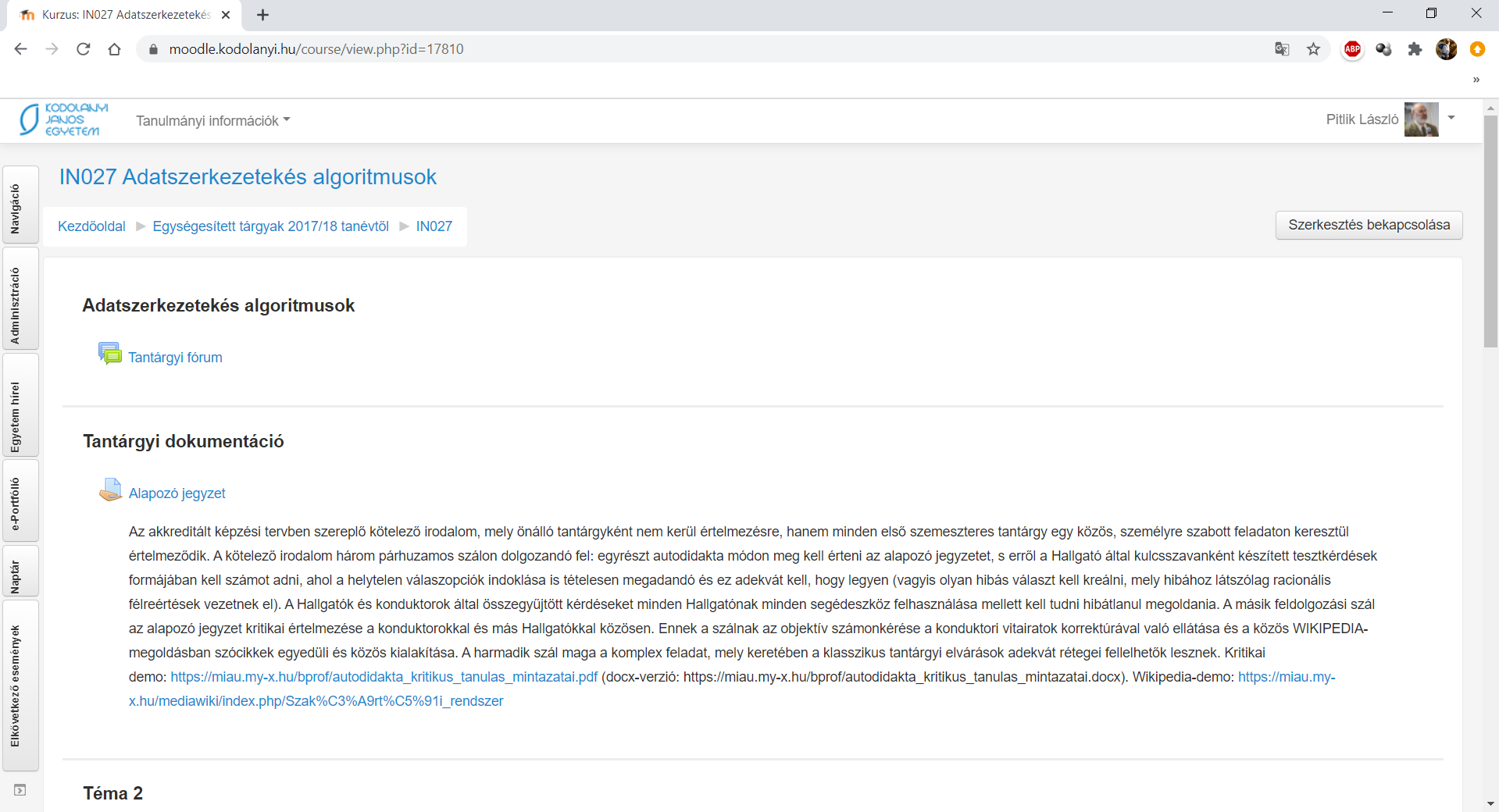
Abstract: This paper tries to deliver new point of views parallel to the old script, in order to ensure the praxis-orientation and a kind of just-in-time-knowledge-management needed for the BPROF-education.

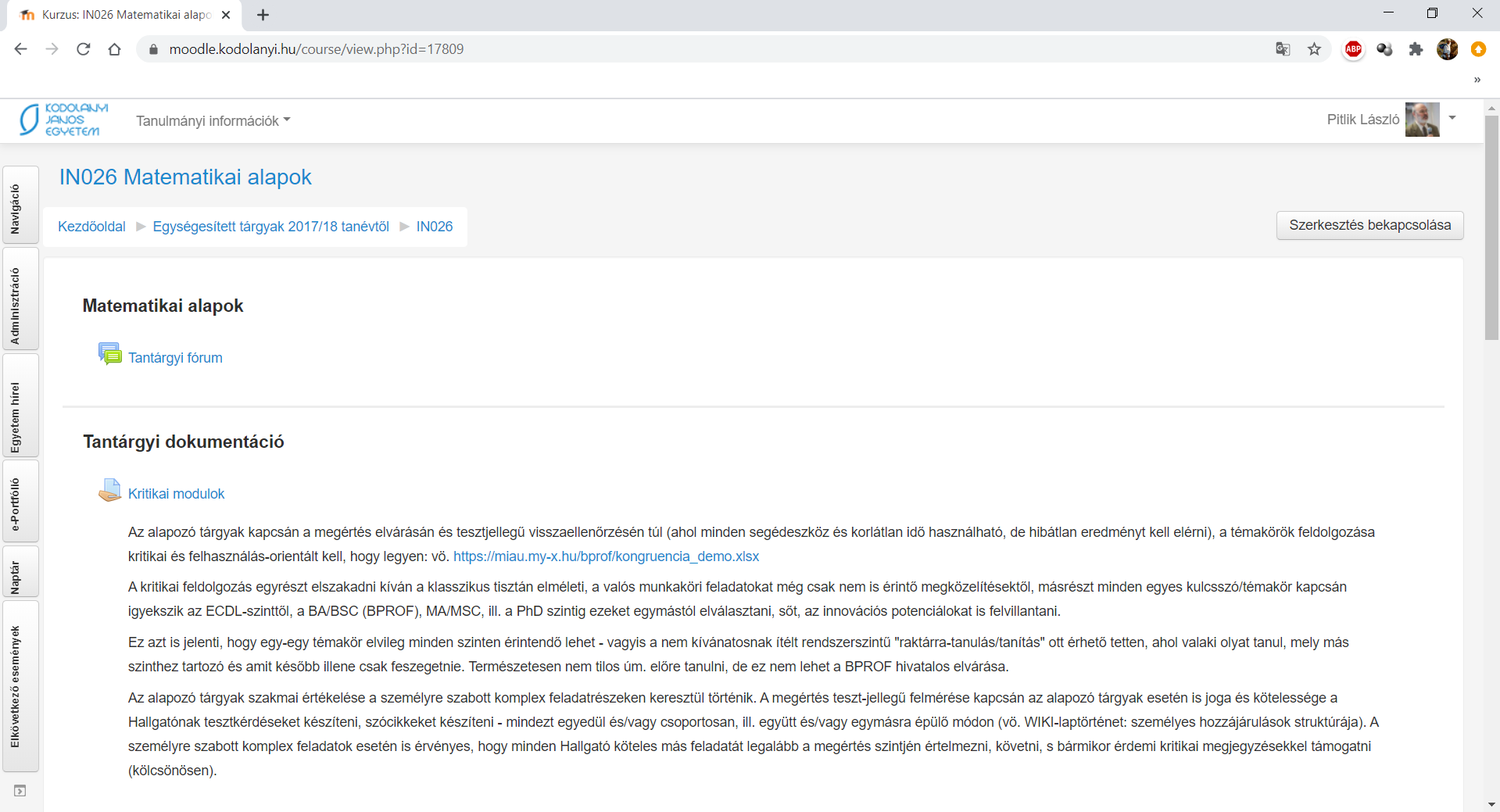
Keywords: knowledge management, didactics, praxis-orientation, dual education, curricula, accreditation, quality management, optimization

# Bevezetés

A cikk apropóját az első rész (<https://miau.my-x.hu/bprof/autodidakta_kritikus_tanulas_mintazatai.pdf>),ill. továbbra is a Házy Attila és Nagy Ferenc (2009): Adatstruktúrák és algoritmusok, <https://www.tankonyvtar.hu/hu/tartalom/tamop425/0046_adatstrukturak_es_algoritmusok/adatok.html> tankönyv és a BPROF képzés akkreditációs anyaga adja.

A speciális BPROF-koncepciót az alábbi tantárgyleírások is igyekeznek érthetővé tenni az érintett Hallgatóság számára: pl.





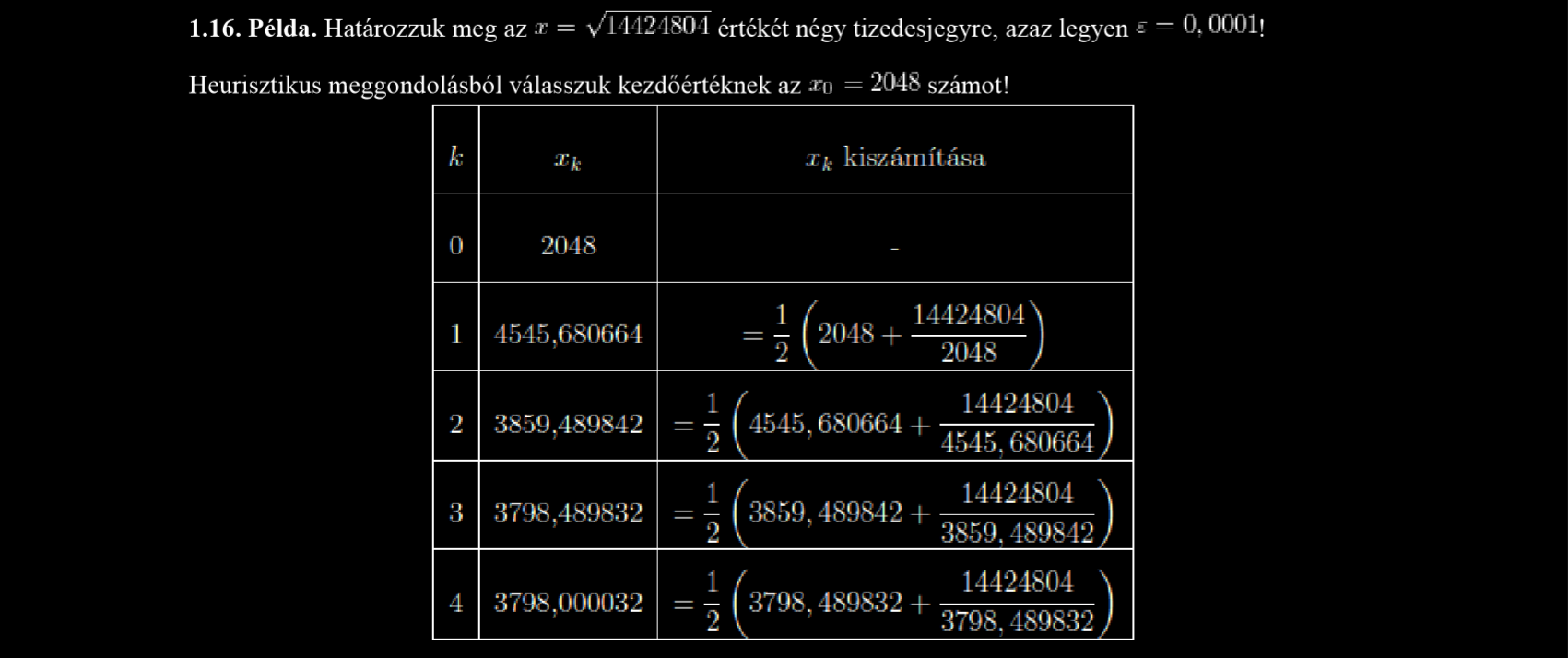
Az első rész az alapozó jegyzet 5.1.5. fejezeténél állt meg, vagyis a második rész innentől (19/20. oldal) folytatja a BPROF-képzés fókuszának optimalizálását. Ahol az első részben az első 20 oldal tekintetében nem volt érdemi észrevétel, azokat a gondolatokat az alapozó jegyzetből a Hallgató köteles megérteni/feldolgozni tesztkérdéssé formálni, ill. beépíteni a személyre szabott komplex, tantárgyi határokat feloldó feladatába.

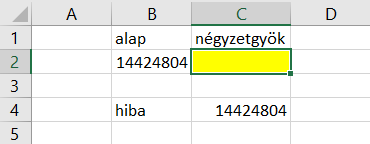
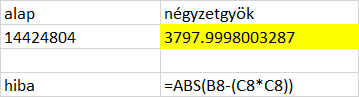
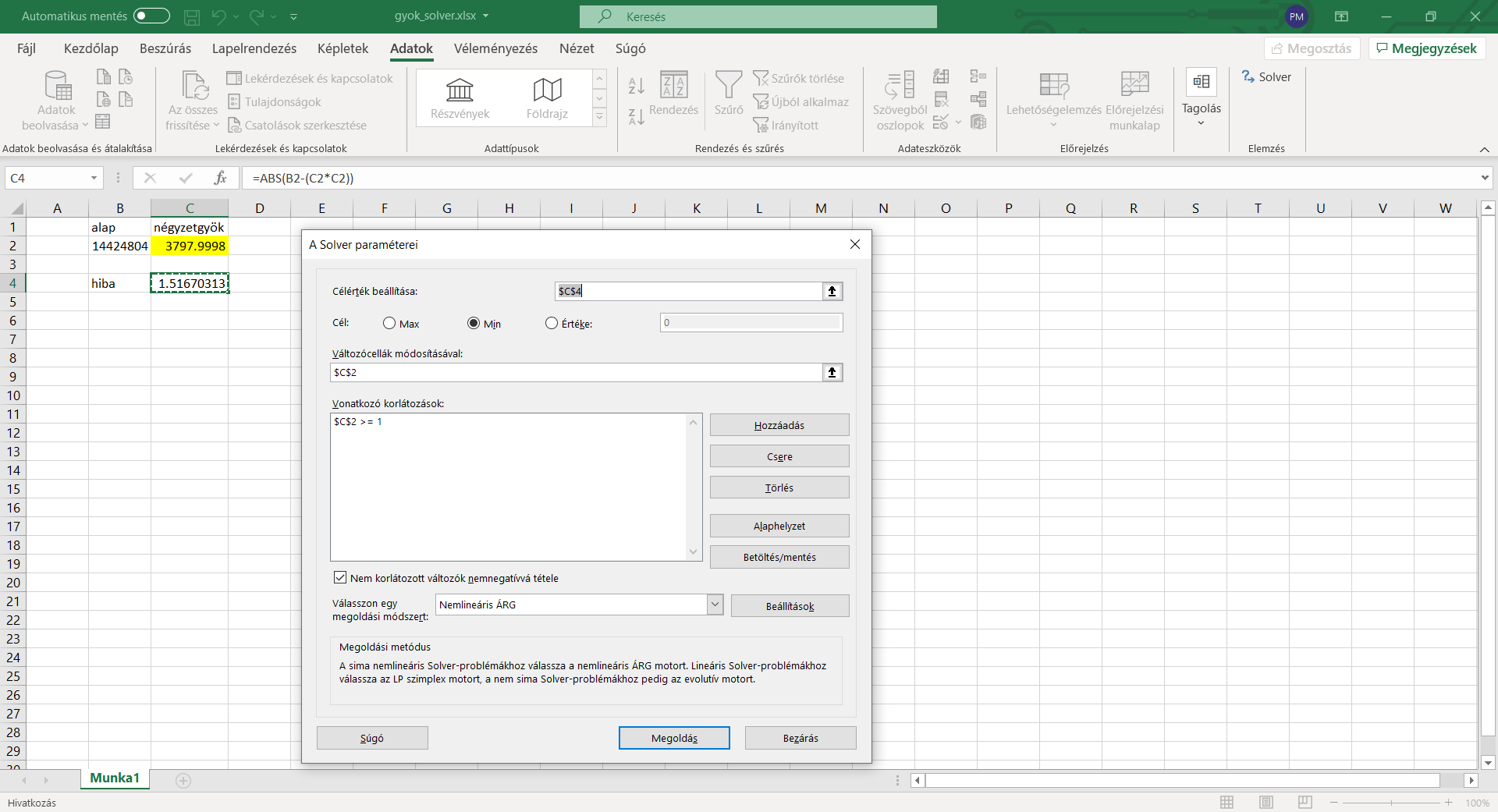
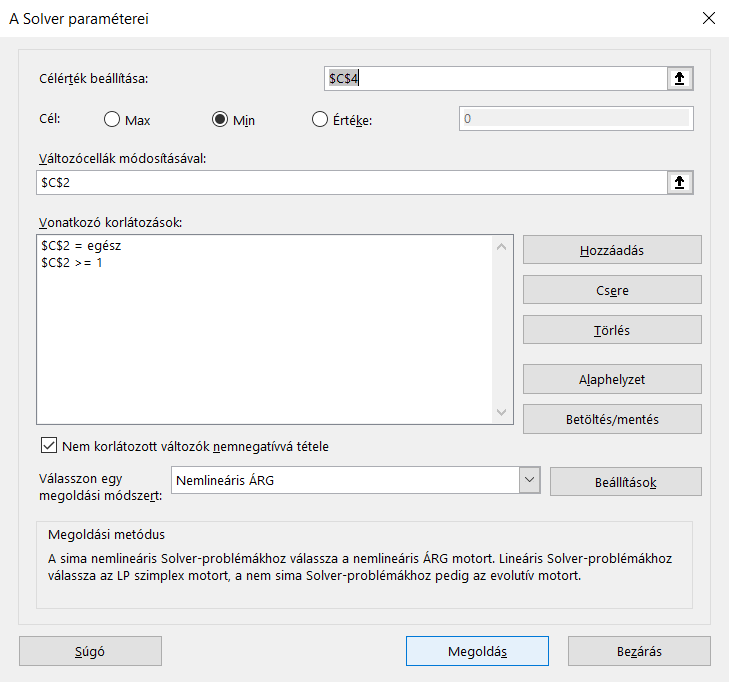
A just-in-time-tudásmenedzsment (<https://hu.wikipedia.org/wiki/Just_In_Time>) lényege, hogy a tanulás a mindenkori problémák mentén és nem raktárra történnek. Természetesen mindenkor ki kell emelni: mindenkinek joga van azt megismerni, megtanulni és raktározni, amit szeretne, de egy racionális képzési rendszer a mindenkori tudáshiányt kell, hogy pótolja, se többet, se kevesebbet.

# Példa – az ember-gép-szimbiózisról

A gyökvonás eltérő megközelítései a programozói gondolkodás megszerzéséhez/kiérleléséhez vezető úton hasznos ujjgyakorlatoknak számítanak. Az alábbi nézet azonban kísértetiesen emlékeztet a gép-ember-szimbiózis eklatáns példájára, az önmagában ismét csak gyakorlatidegen, de mégis az emberi intuíció és a gépi támogatások kapcsolatrendszerét sokszínűen érzékeltetni képes feladatra: a „Melyik az a négy-jegyű szám …?” feladatra: vö. <https://miau.my-x.hu/miau2009/index.php3?x=e0&string=csodak>, ill. <https://miau.my-x.hu/miau/ex-cel-atlon/4jegyuszam.png> (az online könyvtár egyes állományai is a feladathoz kapcsolódnak):

További részletek: <https://miau.my-x.hu/bprof/gyok_solver.xlsx>

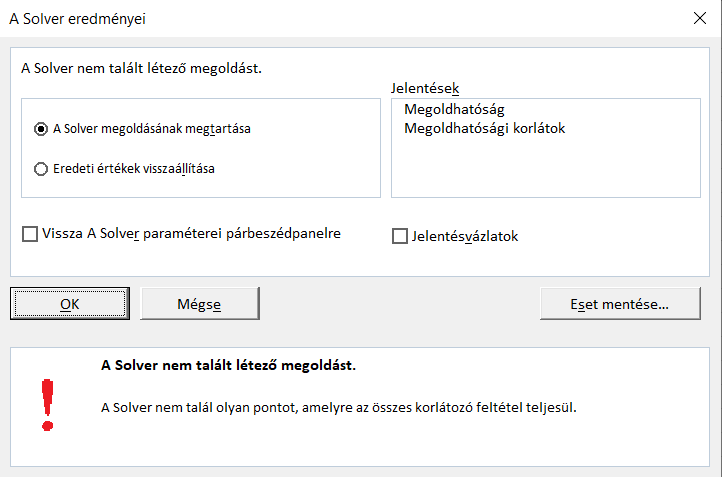
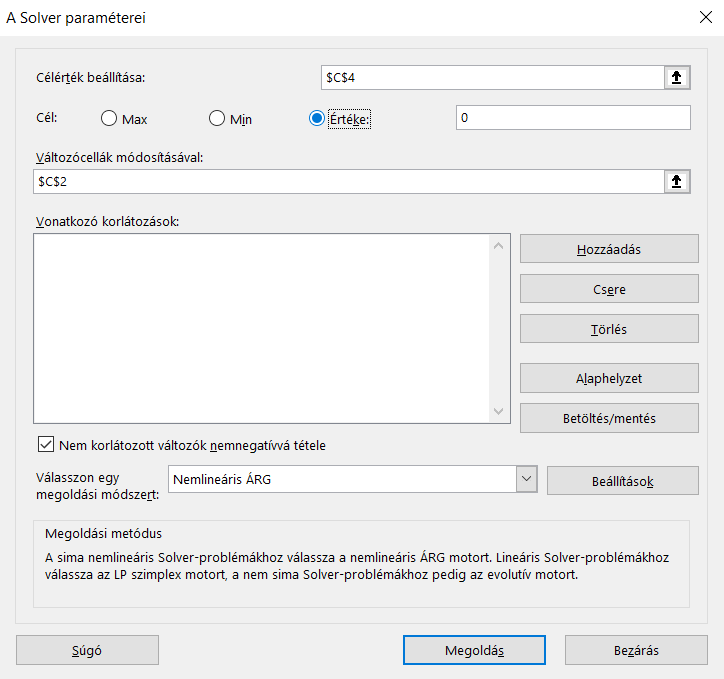
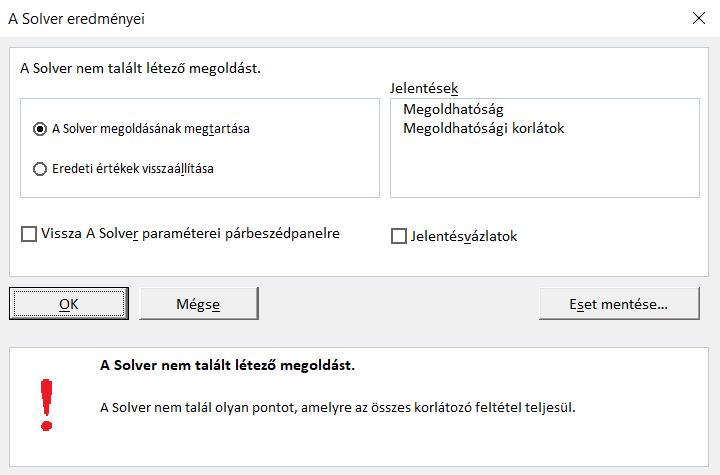


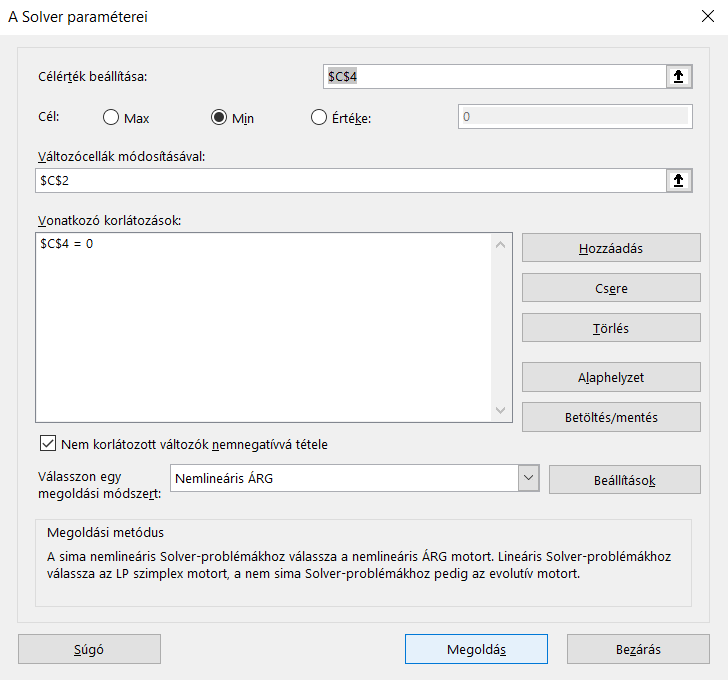
Mint a fenti, a Solver-paramétereket bemutató részlet is világosan jelzi, a négyzetgyökvonás jelensége is úm. solver-esíthető, amennyiben a megoldást kereső nem rendelkezik több matematikai tudással, mint magával a négyzetgyök fogalmával, vagyis azzal, hogy: ha a kiindulási értékből a négyzetgyök értékét önmagával szorozva kivonjuk, nullát kell kapnunk, ill. a nullát tetszőlegesen kell tudnunk közelíteni…

A négyzetgyökvonás tehát egy olyan Solver-alapú feladat, ahol hiba értékét minimalizálni akarjuk és ehhez a gyök értékét módosíthatja a Solver mögötti „engine”. Az ember-gép-szimbiózis és egyben a just-in-time-tudásmenedzsment lényege: az ember csak annyit tudjon, amit feltétlenül tudnia kell, s minden mást bízzon rá a mások által már kialakított gépi megoldásokra – jelen esetben a Solver-re. Az ember tehát megadja a kiindulási (alap) értéket, s megadja a hiba definícióját a minimális/minimalizálnadó matematikai és programozási tudása birtokában, majd megszólítja a Solver-t és elvárja, hogy a mások által már kitalált komplex problémamegoldási (pl. egyenletrendezési, optimalizálási) eszköz elvégezze feladatát.

Mint szinte minden esetben, itt is lehet alternatív megoldásokról beszélni: pl. a hiba értékét nem csak abszolút értékként, hanem hibanégyzetként is lehet értelmezni, ill. a megoldás kapcsán a legjobb megoldástól való pontosság érzékeltethető azzal, ha a keresett (sárga) értékre vonatkozóan elvárjuk az egész-értékűséget (integer-állapotot) is.

Az ember-gép-szimbiózis potenciális sikere azonban nem triviális. Ha nem adunk meg semmilyen kiindulási értéket az ún. sárga cellába, ill. nem adjuk meg korlátozó feltételként azt, hogy a megoldás legyen nagyobb, mint nulla, akkor mindennemű korlátozó feltétel nélkül, a hiba minimalizálását bejelölve a nullát megoldásnak tekinti a Solver. S ha a hiba értékének ideális állapotát NULLA értékűnek állítjuk be (speciális korlátozó feltétel a cél-paraméteren keresztül), akkor pedig azt jelzi a Solver, hogy nem talál megoldást (noha tudjuk, hogy ez létezik):



A Solver-rel való együttműködés tehát speciális trükköket igényelhet, ha a látszólag azonos értékű (nem feltétlenül szükséges részleteket is tartalmazó) megoldások egyike-másika nem vezet eredményre.

A gyökvonás példáját immár nem programozói, hanem Megrendelői szemmel vizsgálva, bármilyen algoritmus kialakítása felesleges/drága egyedi kérdések esetén, ha van Solver-alapú megoldás! Hiszen ez utóbbihoz (szinte) semmilyen matematikai tudás nem kell, vagyis a Megrendelőnek nem kell fizetni ezért, „csak” a megfelelő Excel/Solver (engine) használati tudást kell megfizetni – ami ismét csak a just-in-time-tudásmenedzsment hasznainak egyike. S ez a tudás másodpercek alatt vezet eredményre, s ennek elsajátítása sem igényel többet, mint a normál oktatási menetben az általános iskolától kezdve folyamatosan ismételni, hogy

* Minden feladat megoldásához minden segédeszköz (engine) használható.
* Tudás az, ami forráskódba átírható.
* Az ember-gép-szimbiózis érték.
* A programozás Megrendelői szempontból nem az önmegvalósításról szól, hanem a hatásosságról és a hatékonyságról.
* Minden programozó előbb-utóbb Megrendelő/Felhasználó lesz mások munkájával szemben.
* A raktározott tudás veszíthet értékéből.
* A raktározott tudás új tudást katalizálhat.
* …

Mivel ezek a mondatok egyelőre még a hivatalos tanmenetek szempontjából (így együtt) inkább számítanak eretnekségnek, mint közvetítendő értéknek, így a BPROF-képzés apropóján kell ezeket sulykolás jelleggel továbbadni.

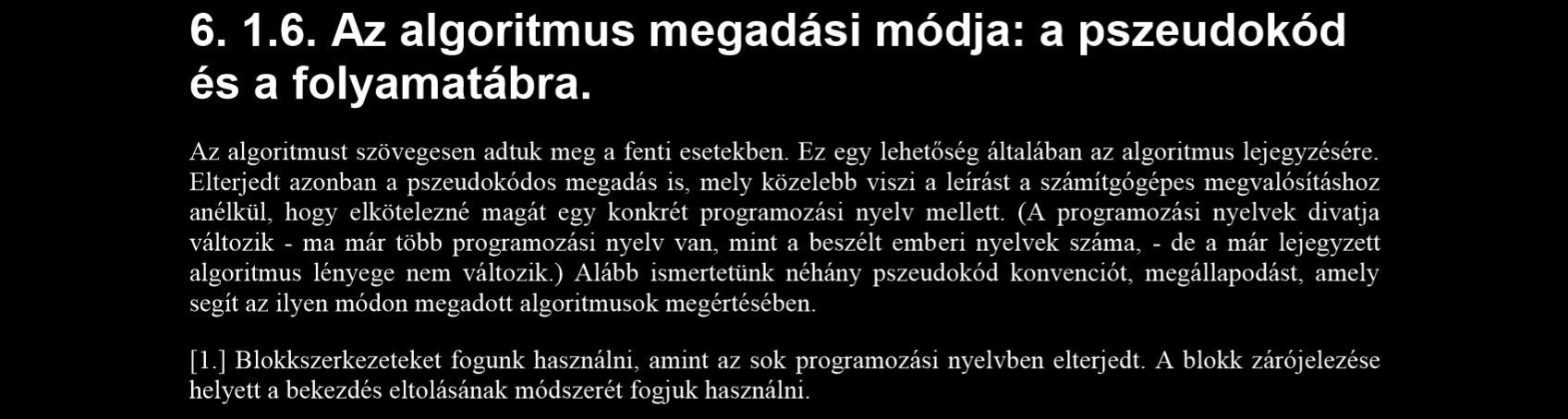
Ezen példa tehát elsődlegesen azt üzeni a leendő üzemmérnök informatikusoknak, hogy törekedjenek az ember-gép-szimbiózisra, mely más módon várja el a kreativitást, mint a klasszikus programozásra való felkészülés. Ez az üzenet nem keverendő össze azzal a célkitűzéssel, hogy programozni már pedig meg kell tanulni minél több nyelven, minél komplexebb kihívások esetén: s nem csak hatásos, de minél több szempontból hatékony kódokat alkotva, ahol a hatékonyságot minél objektívebben kell tudni értelmezni (akarni).

A Hallgatóság szempontjából tehát a Solver-alapúság példaként való említése nem csökkenti, hanem növeli a tananyagot, mert a klasszikus programozási tudás mellé még a Solver-alapúság szemléletét is el kell sajátítani. Ez a többlet azonban társadalmi hasznossággal bír, így kötelező jellege legitim.

Amikor a Hallgató/Programozó válik mások munkájának Megrendelőjévé/Felhasználójává, ahol a munka ellenértékét meg kell fizetni, akkor azonnal beáll a kazohin egyensúly, amikor azért törekszik a Hallgató/Programozó a Solver-alapú megoldásra, ahol csak lehet, mert ő maga is csak ennek ellenértékét akarná kifizetni fordított esetben.

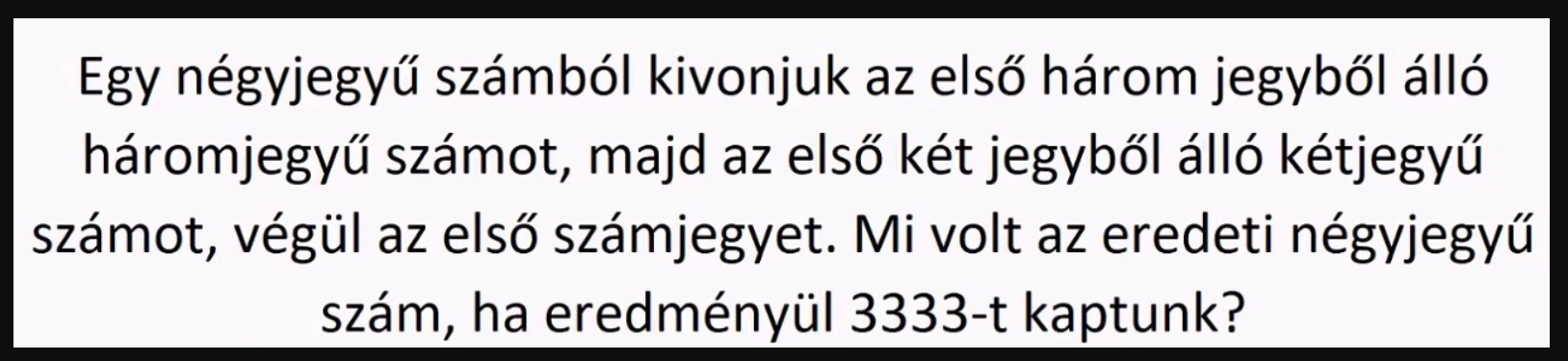
További részletek: <https://miau.my-x.hu/bprof/gyok_solver.xlsx>

# Példa – a szómágiától a tényleges programkódig



A Solver-alapú megoldás esetén a szómágia oldalán csak feladatszöveg van, s nincs klasszikus szöveges megoldás. Ehelyett van a gyökvonás definíciója és ezen definíció hiba-értékké konvertálásának logikája, mely ismét csak létezhet szövegesen (azaz szómágikusan), ahol a szómágia nem valamiféle leértékelése a magyar (más) nyelvű szöveges megfogalmazásoknak, hanem annak a csapdahelyzetnek jellegzetes fogalomként való deklarálni akarása, miszerint a szövegek zömmel félreérthetők, s ebben nem feltétlenül a félreértést produkáló emberi alanyok a hibásak.

Vegyük példaként a fentebb már PNG-ként is idézett Solver-alapfeladatot:



A sok-sok tucatnyi megoldáskísérlet alapján az alábbi félreértési potenciálok mindenképpen felismerhetők voltak a múltban:

* Az első három jegyet milyen olvasásirány szerint értelmezzük, az európai balról jobb, vagyis pl. az arab jobbról balra irány szerint?
* A kétjegyűszámot a négyjegyű és a háromjegyű számok egymásból való kivonásának eredményéből kell-e képezni vagy sem?
* …

Hasonlóképpen szövegértelmezési kihívást jelent az alábbi klasszikus matematika-feladat is:

„Mit kell kérdezni akkor a szembejövőtől egy adott tereptárgy hollétét illetően, ha tudjuk, hogy vannak hazudós szembejövők és vannak igazmondó szembejövő, de mi nem tudhatjuk, mely szembejövő hazudik majd nekünk, s melyik nem, de egyetlen kérdés alapján akarunk egyértelmű választ kapni…” (vö. <https://fejtoro.transindex.ro/index.php?feladvany=18>, ill. <https://fejtoro.transindex.ro/?megfejtes=18>)

„Mit kérdezzen a vándor? Egy vándor útja során egy útelágazáshoz ér, ahol mindig egy őr áll. Tudja, hogy az egyik út a tengerhez vezet, ahova ő elszeretne érni, a másik viszont a hegyekbe visz. Azt is tudja, hogy két őr van, az egyik mindig hazudik, a másik mindig igazat mond, és hogy felváltva egyik nap egyik, másik nap a másik őrködik. De azt nem lehet tudni, hogy mikor ő odaér, éppen melyik az ügyeletes. Mit kell kérdezni az őrtől, ha csak egy kérdést tehet fel neki, és azt szeretné, hogy az a tenger felé vezető utat mutassa meg neki? Megfejtés - a Transindex.ro portálról

A megfejtés Mit kérdezzen a vándor? Azt kell megkérdeznie az ügyeletes őrtől, hogy merre mutatna a másik őr, ha megkérdezné tőle, merre vannak a hegyek? Ha az igazmondó őr van ott, akkor az tudja, hogy a másik hazudna, ezért ő a tenger irányába fog mutatni. Ha a hazug őr az ügyeletes, akkor az is tudja, hogy a másik őr a hegyek felé mutatna erre a kérésre, de mivel ő hazudik, szintén a tengert fogja iránynak megadni. - a Transindex.ro portálról”

Ez tehát a klasszikus megfejtés.

De miként lehet levezetni azt, hogy pl. az alábbi szöveges (szómágikus) megoldás helyes-e?

Potenciális megoldás: Azt kell kérdeznie a vándornak az őrtől, vajon igazat mondana-e, ha azt állította volna előzőleg, hogy a tenger pl. balra van?

A potenciális megoldás alapötlete a tagadás tagadásának kiaknázni akarása. A kérdés, sikeres-e ez a kiaknázási kísérlet? S hogyan kell bizonyítani a sikerességet/sikertelenséget?

1. Ha az őr igazmondó és a tenger balra van, akkor igaz lenne az első (burkolt kérdésre vonatkozó) állítás, hogy a tenger balra van, s igaz lenne, hogy igazat mondott volna ezzel, tehát a tenger balra van.
2. Ha az őr igazmondó és a tenger jobbra van, akkor a burkolt állítás nem létezik, mert az igazmondó nem hazudott volna, így a nem létező állítás igazságtartalma nem vizsgálható, tehát az igazmondó őr lefagy és csak a vállát vonogatja, vagyis ebben a lefagyásként leírható esetben az esetben a tenger jobbra lesz…
3. Ha az őr éppenséggel hazug és a tenger tényleg balra van, akkor a hazug őr fagy le, hiszen nem mondhatott volna igazat elsőre, így nincs miről hazudnia. A tenger azonban ezen lefagyáskor balra van, ami máris ellentmond a 2. pontnak.
4. Ha az őr hazug és a tenger jobbra van, akkor az első burkolt állítás az elvárásoknak megfelelő hazug, de ezt nem mondhatja ki a hazug őr, tehát azt fogja állítani, hogy a burkolt állítás igaz volt – de ez az igaz válasz és az 1. pont igaz válasza ellentétes tengerirányokat jelent.

Vagyis pl. a két leblokkolás ellentétes tenger-irányt jelent. S hasonlóképpen a két nem leblokkolós esetben a tenger helyzete ismét ellentmondásos.

Abban az esetben, ha nincs leblokkolás, s a lefagyás okát ignoráljuk, akkor a 2. esetben az igazmondó az egyébként tőle nem elvárható hazugságról azt fogja mondani, hogy hazugság – tehát a tenger jobbra van.

A 3. esetben – ismét csak nem foglalkozva azzal, hogy a hazug őr nem mondhatott volna igazat, a burkolt (az őrnek tulajdonított) állítás (= tenger balra) esetén az őr azt kell, hogy mondja, hogy ez hazugság.

A két revidiált eset, amikor a válasz hazugság volt azonban ismét ellentétes tenger-pozícióra utal, tehát használhatatlannak minősül.

A lefagyás a 3. esetben úgy is feloldható, hogy a hazug őr arra a neki tulajdonított feltételezésre, miszerint a tenger balra van, azt is reagálhatja, hogy ha ő mindig hazudik, akkor a neki tulajdonított, soha előállni nem tudó állítás is hazug volt, tehát ahhoz, hogy most is hazudjon (tagadás tagadása, hazugságlánc), ezen állítást igaznak kell beállítania – amivel visszakapjuk, megerősödik a tenger valódi (bal) iránya. S ez megfelel a számozott 1-es esetnek! De ez az interpretáció ellentmond a számozott 4-es esetnek, pl. mert a blokkoláshoz is vezethető ok folytán már nem is egységes maga az őrben futó rendszer/szoftver.

Vélelmezhetően a szómágia még tovább lenne képes gazdagítani a fenti eseteket. A kérdés már csak az, vajon tényleg hibás a fentebb többszörösen is hibásnak beállított megoldás? Illetve, ha a fenti interpretációk helyesek, akkor igaz-e, hogy nincs helyes értelmezésre vezető szómágia ezen a szálon?

# Példa – pszeudo-kód, mint a gondolkodás alapja?!

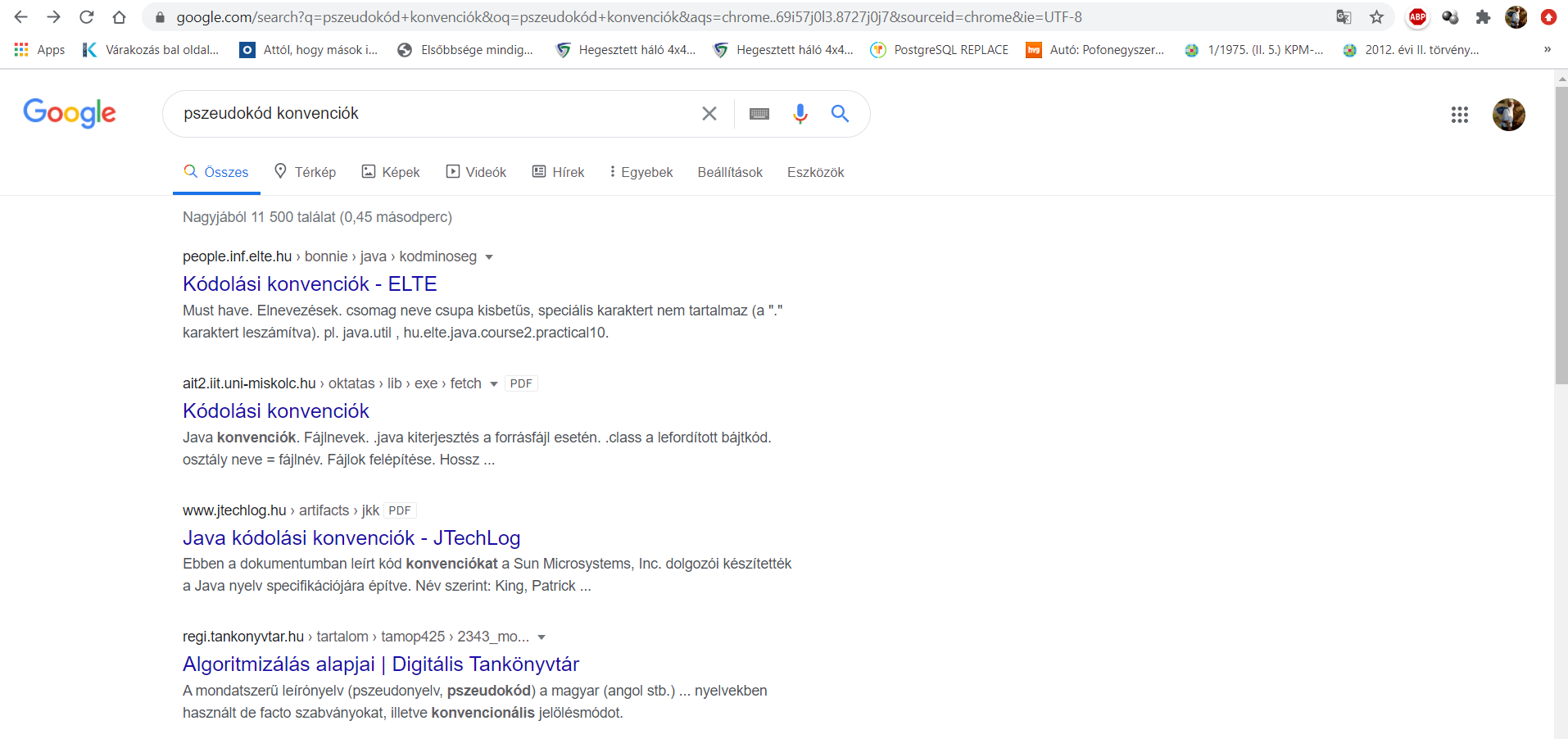
Folytatva az utolsó inverz-kép értelmezését a pszeudo-kód kapcsán:

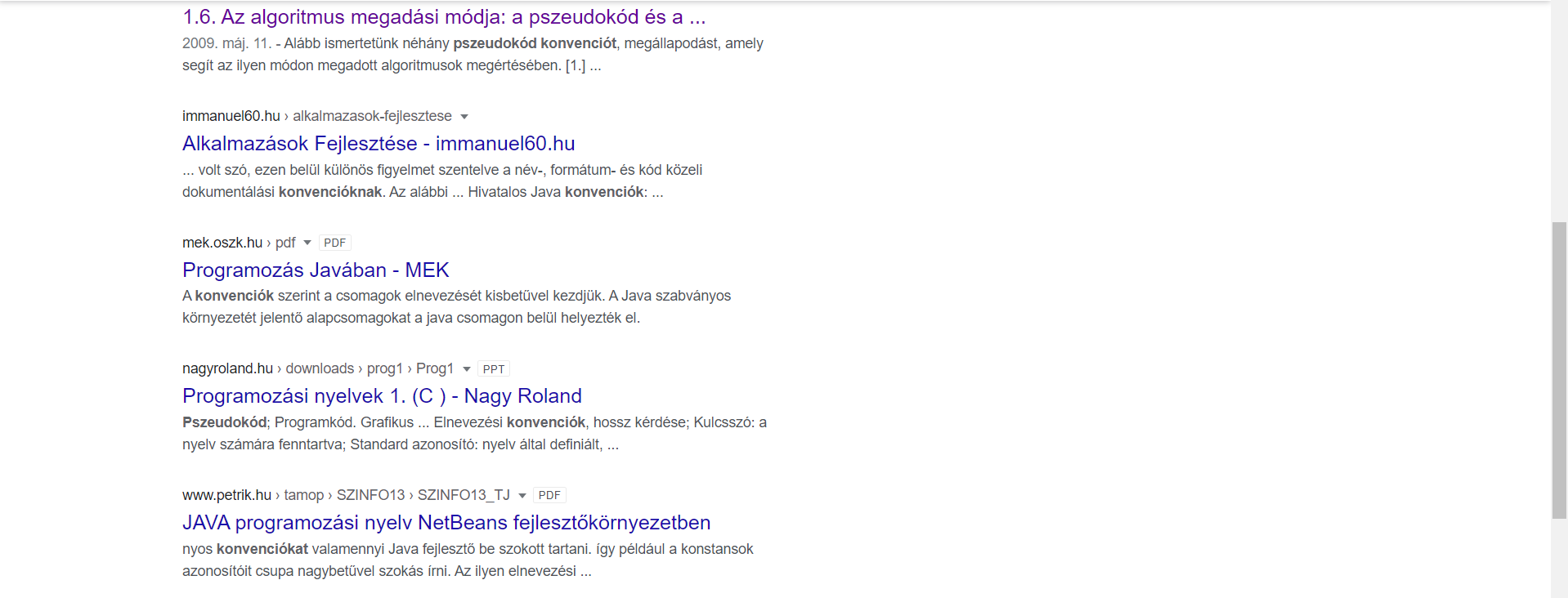
Már az a tény, hogy több pszeudo-kód konvenció létezik, világosan utal arra, hogy a problémák emberek általi értelmezése, az emberi agyak működése és a probléma-ember-interakció komplexitása indokolni látszik többféle megközelítés létjogosultságát, ahol sok probléma és sok ember eredőjeként talán minden konvencióról belátható lenne, hogy átlagosan ezek mind egyformák, s mégis igaz kell, hogy legyen, hogy adott embertípus (gondolkodásmód) és/vagy adott problémák kezelésekor egyes konvenciók előnyösebbek, mint mások – különben nem illett volna, hogy kialakuljanak ezen alternatívák.

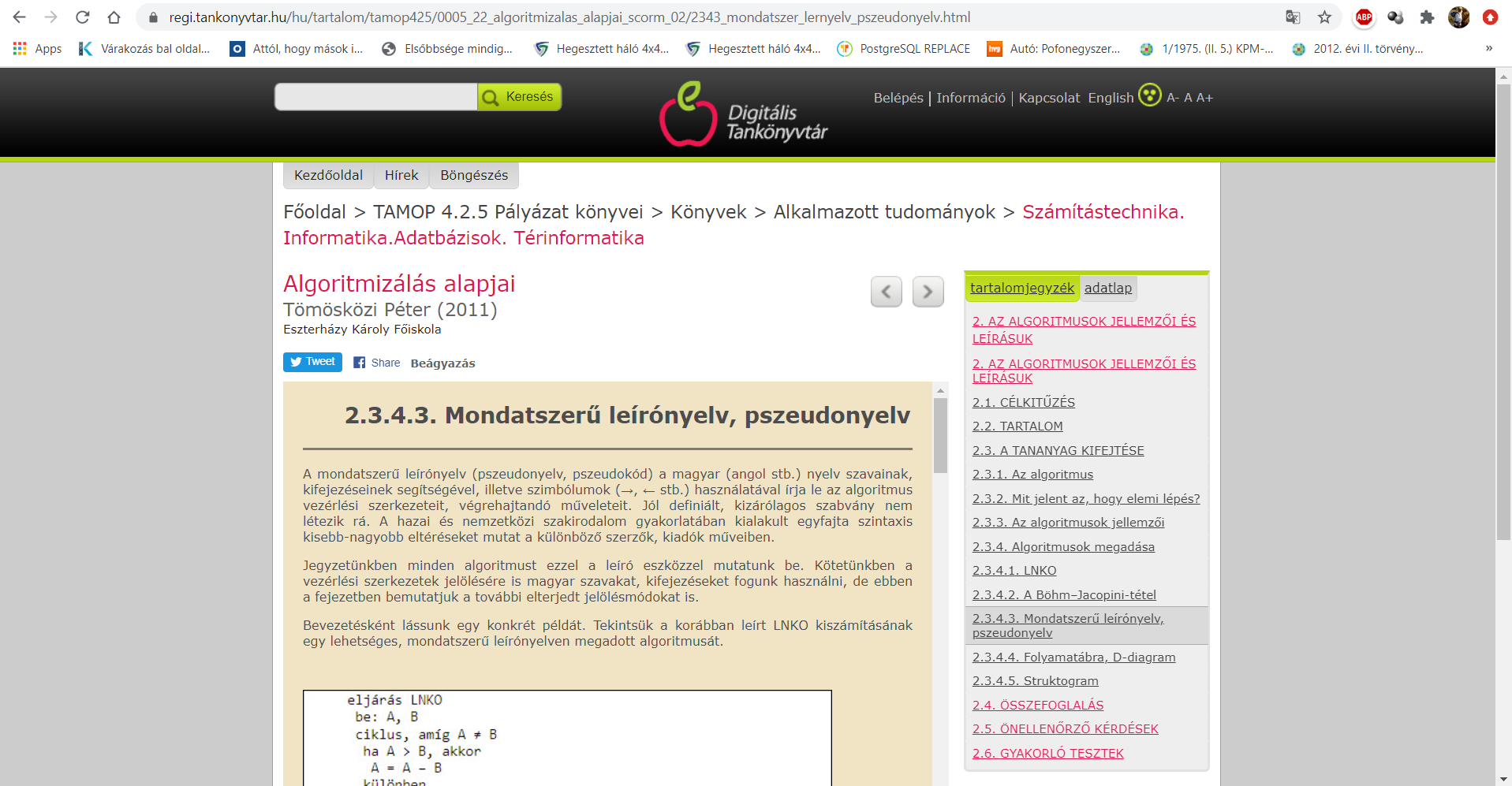
A blokk-szerkezetre utalás, de különösen ezen belül a zárójelezés helyett a bekezdés eltolás, mind-mind az alternatív konvenciók (konvenció-variánsok) paraméterei.

Ideális esetben a konvenciók rendszertanának oktatása felmerülhet Hallgatói igényként, ill. oktatói javaslatként. Ha azonban rákeresünk a pszeudo-kód + konvenciók kifejezéspárra, akkor magyar nyelven nem találunk azonnal ilyen jellegű rendszertani utalásokat. Mint az az alábbi találati lista alapján megállapítható, egyetlen egy találat foglalkozik a jelenséggel megfelelő általánossággal (vö. <https://regi.tankonyvtar.hu/hu/tartalom/tamop425/0005_22_algoritmizalas_alapjai_scorm_02/2343_mondatszer_lernyelv_pszeudonyelv.html>), s ennek a találatnak a tételes tartalma az alábbiakat állítja:

* „Jól definiált, kizárólagos szabvány nem létezik rá.
* A hazai és nemzetközi szakirodalom gyakorlatában kialakult egyfajta szintaxis
* kisebb-nagyobb eltéréseket mutat a különböző szerzők, kiadók műveiben.”

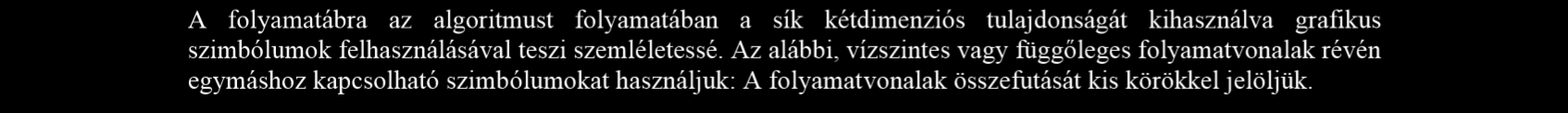


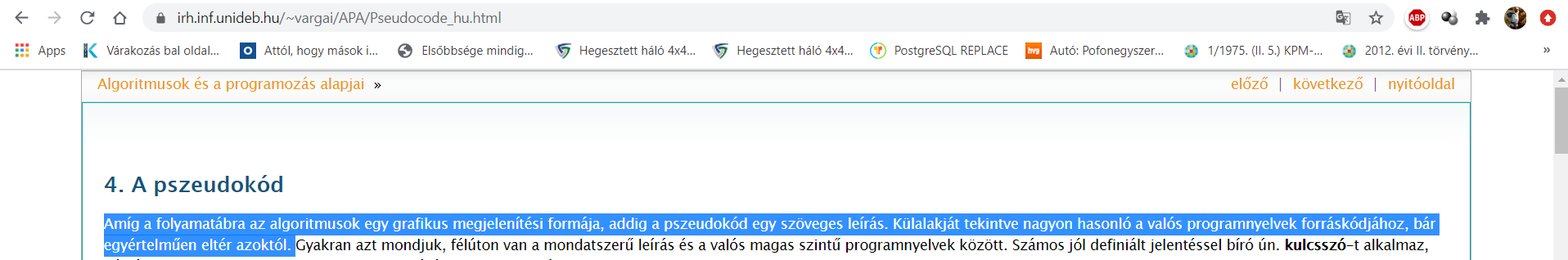




A pszeudo-kód tehát olyan lingua franca (<https://hu.wikipedia.org/wiki/Lingua_franca>), melynek már úm. tájszólásai is vannak. Más analógiát használva: a pszeudo-nyelv egy fajta műszaki rajz, mely alapján a gyártás precízen illene, hogy lefusson. A pszeudo-kód létének értelme triviális, a klasszikus szómágiánál kevesebb félreértésre ad lehetőséget! De a jelek szerint a pszeudo-kód alapján automatikusan szoftvert készíteni nem tűnik reális kihívásnak – ami ütközik a KNUTH-i elvvel.

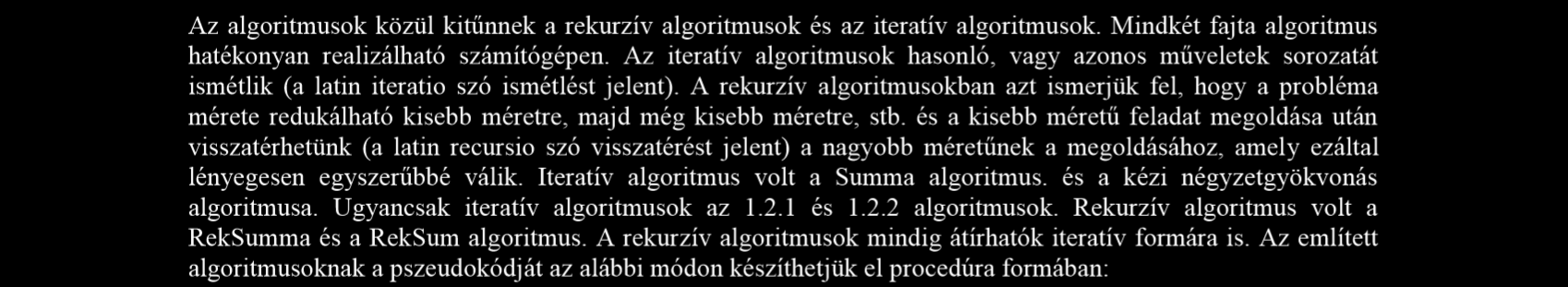
# Példa – Folyamatábra-alapú gondolkodásmód





A folyamatábra a fenti idézet alapján (vö. <https://irh.inf.unideb.hu/~vargai/APA/Pseudocode_hu.html>) nem feltétlenül értelmezendő pszeudo-kódként, mégis igaz: quasi azonos funkciót tölt be (vö. műszaki rajz, mint analógia). A grafikusság nem releváns attribútuma az átvinni szükséges információknak, bár a felhasználók számára egyéni preferenciák kialakítására ad teret. A folyamatábrák esetén is elvárás lenne, hogy ezekből a tényleges programkód (bármilyen nyelven) automatikusan legyen képes keletkezni. Sőt, az is elvárható hosszabb távon, hogy a természetesen emberi nyelven megfogalmazott szoftver-funkcionalitások nyomán szoftver tudjon keletkezni, még ha ehhez a konvertáló keretrendszernek vissza is kell kérdeznie, hiszen két ember kommunikációjának eredményeként a szoftver elő fog állni. A mesterséges intelligenciák pedig az ember képességeit akarják a számítógépek számára átadni.

# Példa – Algoritmusok „rendszertana”



Alapvetően a rendszertan, vagyis a sokelemű halmazok leírása a tudomány olyan speciális vadhajtása (vö. növény-rendszertan, állat-rendszertan, gombahatározó), mely egyszerre bizonyos értelemben káros és didaktikai (knuth-i) szempontból hasznos.

Káros a rendszertan-alapú gondolkodás, mert zömmel akkor is besorol egyedeket halmazokba, ha a halmazok hibásak, a besorolási elvek kiforratlanok. Vagyis a rendszertanok úm. fecsegnek, s nem merik általában véve felvállalni, hogy a rendszerezés elvei esetlegesen nem is léteznek (még) kellően stabilan. A rendszertanok (vö. emberek rendszerezése) pedig a történelmi tapasztalatok alapján a ma oly gyakran emlegetett „P.C.”-től (political correct) inkább a rasszisztikus megoldások irányába hat. Másrészt a PC olyan önkorlátozásokat vállal fel, mely végül már a szómágiát magát is zavarba hozza, mert tiltott kifejezések okán egyre zavarosabb virágnyelvi verziók alakulnak ki.

A rendszertanok jó tulajdonsága, hogy szocializálják a befogadókat és terjesztőket a szakértői rendszer jellegű gondolkodásmódra. A gombarendszertan így nem azonos a gombahatározóval, hiszen: teljesen lényegtelen, ha egy adott gombáról csak annyit akarok tudni, vajon ehető-e, vagy sem, hogy tudom/tudhatom, milyen törzs-osztály-rend-család-nemzetség-faj besorolást kellene, hogy kapjon.

Az algoritmusok kapcsán is lényegtelen végső soron, milyen (rendszertani) leíró jellemzői vannak egy-egy objektumnak, ha azt teszi, amire létrehozták! Másrészt: az iteráció és/vagy a rekurzió olyan alapvető karakterisztikák, amiket nem árt a programozás előkészítése kapcsán deklarálni. Ennek ellentmondani látszik a fenti idézet utolsó előtti mondata: „A rekurzív algoritmusok mindig átírhatók iteratív formára is.” Ahol a konvertálhatóság egyik típusból a másikba, formálisan feloldja a típusok létjogosultságát. Másrészt azonnal felvetődik, hogy az indokolt rekurzív megoldások iteratívtól való eltérésének előnyösségét miként lehet bizonyítani, kimutatni?

A rendszertanok arra mindenképpen jók, hogy segítenek felismerni a rendszernek vélt valami zavarait. Példaként itt érdemes megemlíteni a következő problémát és az erre adott szakértői reakciókat:

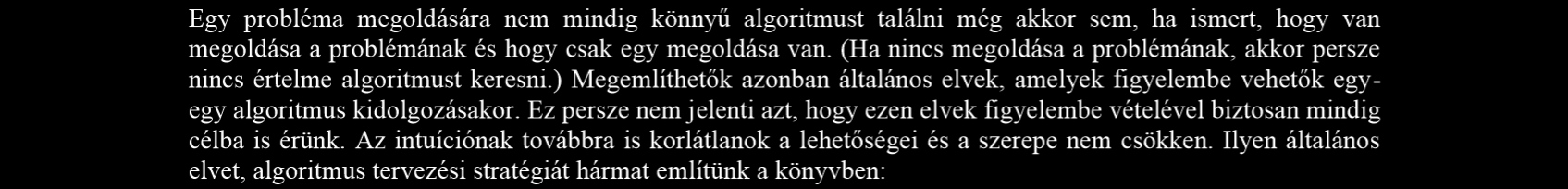
Létezik-e olyan webes katalógus/táblázat/adatbázis, melyben a sorfejlécen a növények nevei állnak, az oszlopfejlécen pedig minden, a növénytanban ismert, a növények leírására alkalmas jelenség neve? (A táblázati cellák értelemszerűen az adott növény adott szempontból felvett tulajdonságértékei lennének = OAM = objektum-attribútum-mátrix). Ezen táblázat alapján meg lehetne pl. mondani egy-két szűréssel, vajon hány olyan növény van, mely egylaki és egyivarú és egyszikű és ésésésésés (ha van ilyen egyáltalán)? Miért fontos ez? Mert az érettségire készülők egyike-másika magától tapintott rá arra korábban többször is, hogy egy ilyennek lennie kellene már réges-régen, de senki nem lelt rá még erre…

A szakértői válaszok [és az ezekhez fűzött egyéb kommentárok]:

Ilyen listáról nem tudni, annál is inkább, mert a Földön előforduló növényfajok számát akár 500 ezerre is teszik. Ennélfogva a listát mindenképpen be kell szűkíteni térben és rendszertani kategóriák tekintetében is. [500.00 rekord ma már elférne egy xlsx táblázatban, vagyis (némi áthallással) "nem a méret a lényeg?!] Számunkra, Magyarországon, kézenfekvő, hogy csak a hazai (edényes!) flóra mintegy 2400 fajával foglalkozzunk. [Ez a méret pedig már egy-két ember számára is kezelhető méretű kihívás!] Arról sem tudni, hogy ezek listája létezne az Interneten. De nem lehetetlen azonban… (pl. <http://www.botanikuskert.hu/node/644>) [A QR kódok mögötti információk létezhetnek esetleg már adatbázisban?!] Volt, aki 10-15 éve - a régi növényhatározó felhasználásával - begépelte az abban szereplő összes fajt egy táblázatba, úgy, hogy a sorfejlécen a fajnevek szerepelnek. A táblázatnak 4 oszlopa volt, ezek a tudományos és magyar fajneveket, valamint családneveket tartalmazzák. Mást nem. Ez a fájl ma is létezik. [Szabadna ezt egy hazai ifjú-botanikusok mozgalmává átalakítani és az alaptáblázatot webre téve arra kérni minden érdeklődőt, hogy írja be azt, amit tud, s adja meg hozzá azt az egy-két webes URL- referenciaként, ami bizonyítja állítása igazságát?! Sőt, ez lehetne egy wikipedia-s megoldás is...] A harasztokat, nyitvatermőket, egyszikűeket és kétszikűeket kevés munkával be lehet vinni a táblázatba, a családnevek helyettesítésével. Ez mindössze 1-2 órai munka. A beporzással és más tulajdonságokkal kapcsolatos adatok viszont nincsenek összegyűjtve. [Éljen a "kollaboratív" munka!?] Van azonban néhány probléma a táblázattal és az elvárt beporzási adatokkal is. A szóban forgó táblázat a régi, 2000-es kiadású határozókönyv alapján készült. Azóta a fajok 10-15%-ának a magyar és a tudományos neve is (a gén-szekvenálások adatainak multivariációs analízissel való összehasonlításának eredményeképpen), sőt a fajok rendszertani besorolása is megváltozott, így ez a lista kemény revízióra szorul. [vö. mit ér a rendszertan, ha dinamikusan változik? – illetve: erre is jó lehet a közösség ereje?!] Közben megjelent egy új magyar határozókönyv is, de az ebben használt fajnevek egy része nem egyezik meg a Flora Europaea adatbázisában találhatóval. Ennek több oka van, amit most nem érdemes részletesen taglalni. A lényeg az, hogy egy ilyen lista készítésekor el kell dönteni, melyik adatbázist használja az ember. Akár be is lehet vinni a táblázatba, hogy egy-egy fajt a Flora Europaea nem ismer el, [Igen: ez a konszolidációs kihívás nem nagy kihívás - műszaki szempontból...] ill. más néven használ vagy máshová sorol be. Egy ilyen összehasonlítás elvégzése önmagában nem kis feladat lenne. A beporzással az a gond, hogy vannak jócskán vegyes beporzású fajok is, s a beporzás módjainak megoszlása is fajonként, sőt térben és időben is eltérő lehet. [Akkor majd ezt írjuk a megfelelő cellákba!?] …

Még egyszer hangsúlyozandó: a probléma olyan probléma/kérdés, melyet érettségin, OKTV-n, esetleg show-műsorokban bárhol a világon már nem egyszer tettek fel az érintetteknek! Nem szabad elfelejteni itt sem: tudás az, ami forráskódba átírható… s forráskód minden OAM (objektum-attribútum-mátrix) is…

# Példa – Az intuíció szerepe



A KNUTH-i elv értelmében tudás az, ami forráskódba átírható. A fenti idézet értelmében az emberiség még nem jutott el arra a szintre, hogy a felismerni vélt probléma (szöveges) leírása alapján a megoldást jelentő forráskódot forráskód állítsa elő. Ezért van szükség programozókra, akik agya képes a probléma szöveges/pszeudo-nyelvi/hibrid leírása és a megoldás forráskódja közötti áthidaló szerepet hatásosan és hatékonyan betölteni.

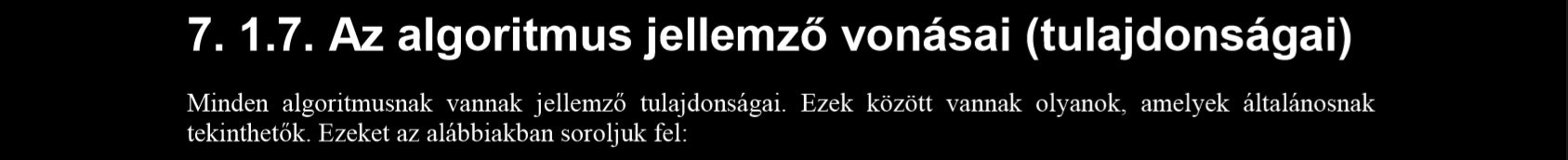
Itt érdemes visszautalni a Solver-alapúság matematikai tudást minimalizáló voltára, mely azonban nem oldja fel az emberi célirányosság, szándékosság szükségszerűségét, vagyis a Solver megszólítani tudását. A Solver-alapú gondolkodás quasi ugyanazon emberi intuíciót igényli, mint az algoritmizálás klasszikus útja. S az emberi agy képes is arra, hogy az ember-gép-szimbiózist eltérő formákban álmodja és valósítsa meg.

A fenti idézet utal három stratégiára:

* „Oszd meg és uralkodj
* Mohó algoritmus
* Dinamikus programozás”

Az operatív megvalósulás szintjén ezek hibridizálása, keverése, egymás utáni tetszőleges gyakorisággal való elhelyezése, váltogatása jelenti a programozás mindennapjait, így ezek a típusok is a rendszertan kapcsán mondottak szerint értelmezendők: egyszerre hasznosak és feleslegesek a maguk módján.

# Példa – Az algoritmus is objektum



Az algoritmusok olyan objektumok, melyeknek számos attribútuma lehet, s így az algoritmusok, éppúgy, mint emberek, egyéb élőlények, tárgyak, folyamatok, szervezetek/intézmények, stb. alávethetők az objektumok összehasonlítására támaszkodó eltérő elemzéseknek. Az algoritmusok kapcsán az egyik triviális célja ezek összevetésének: a melyik a legjobb kérdésre adandó válasz algoritmikus levezetése tetszőlegesen sok tényező alapján, ill. adott objektum adott attribútumértékének becslése ennek hiánya esetén. Más megfogalmazásban: minden algoritmus alapján ennek leíróit figyelembe véve anélkül kellene tudni hatékonysági mutatókat becsülni, hogy ezek ténylegesen mérésre kerülnének (vö. Dobó Andor: hasonlóságok hasonlósága, ahol a feladat az, hogy egy mátrix minden hiányzó elemét pótoljuk számítással, amennyiben a táblázat egy sora és egy oszlopa teljeskörűen kitöltésre került kiindulási állapotként).

Az algoritmusok egyes attribútumai nem függetlenek a környezeti feltételektől (pl. egy algoritmus futásgyorsasága hardver-függő illik, hogy legyen).

A fenti idézet által sejtetett attribútumok a következők:

* „Kiinduló adatok lehetséges halmaza
* A lehetséges eredmények halmaza
* A lehetséges közbülső eredmények halmaza
* A kezdési szabály
* A közvetlen átalakítási szabályok
* A befejezési szabály
* Az eredmény kiolvasási szabálya”

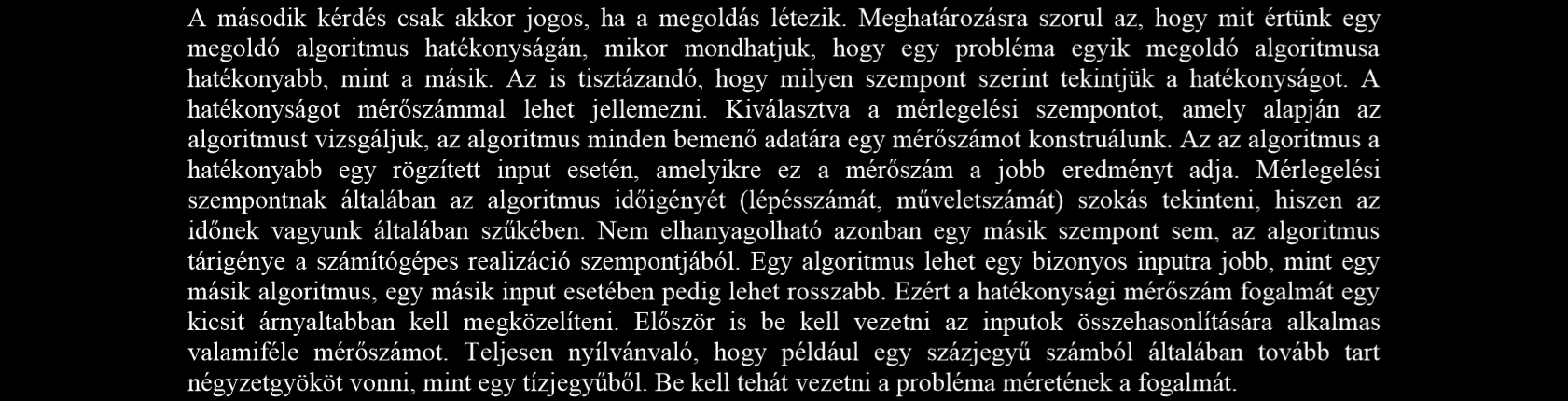
Míg az alapozó jegyzet a gyökvonás példáját mutatja be, addig itt és most a fentebb bevezetett többtényezős automatikus objektumértékelési rendszer, vagyis a hasonlóságelemzés példája kerül nagyító alá annak érdekében, hogy a gép-ember-szimbiózis Solver-alapúsága is teret kaphasson:

A hasonlóságelemzés alapjai: <https://miau.my-x.hu/mediawiki/index.php/Hasonl%C3%B3s%C3%A1gelemz%C3%A9s>, ill. <https://miau.my-x.hu/miau/196/My-X%20Team_A5%20fuzet_HU_jav.pdf>

A hasonlóságelemzés a fenti háttérinformációk alapján tehát:

* Pozitív egész számokból (rangsorszámokból, egyéb Y-értékekből) indul ki.
* S outputja is célszerűen pozitív egész számokból áll (vö. normától való eltérés, valós Y-tól való eltérés adott kerekítési logika alapján).
* A közbülső eredmények kapcsán beszélni illik egyrészt a lineáris programozás szabványos egyenletrendszereiről, mely állományok értelemszerűen szövegnek minősülnek, s beszélni illene a meghívott solver-engine részeredményeiről, melyek közül az igazán közbülsőek ezen algoritmusok zártsága miatt nem feltétlenül ismertek, de a solver-engine outputja ismét csak text, ami lehetne csak pozitív egész számokból álló lista is, amennyiben a listaelemek jelentése előre kódolva lenne.
* Kezdési szabályról (vagyis például arról, melyik attribútummal kell kezdeni az OAM feldolgozását) csak önkényes döntést lehet hozni: pl. az elsővel (balról jobbra), mert ennek nincs szerepe a solver-engine-t megszólító egyenletrendszer, mint részeredmény kialakításában.
* Az átalakítási szabályok kétrétegben értelmezhetők: egyrészt az egyenletrendszerig való eljutás kapcsán FKERES()-jelleggel kezelni kell az attribútumonkénti azonos inputértékeket, ill. le kell vezetni az egyes objektumokra vonatkozó becslések és a tények összevetését támogató nézeteket.
* A befejezési szabály fogalma pedig kinyitja Pandóra szelencéjét: a befejezési szabály ugyanis lehet szimpla ellenőrzése annak, hogy létezik-e egyáltalán a solver-engine válasza, de lehet egy végtelenül komplex hermeneutika, mely azt vizsgálja, mikor szabad a technikailag létező választ rendszerválasznak tekinteni (vö. fecsegő modellek: pl. trendszámítás, melynek minden jövőbeli időpontra vonatkozóan van véleménye, noha az alapját adó lázgörbe alapján világos és előre belátható, hogy tetszőlegesen sok növekedés nem következhet be soha egymás után az ismert mintázatok alapján, mégis lehet eredménynek tekinteni a szigorúan monoton növekedést az egymást követő időegységekre vonatkozóan. Lehetne azonban vizsgálni, vajon mely növekedési monotonitás a legkevésbé valószínű, s ekkor jobb híján a trend maga nem fecsegne, hanem felvállalná a rendszerszintű nem-tudom-választ. A hasonlóságelemzés esetében a konzisztens végeredmény ténylegesen végtelen komplexitással vizsgálható: vö. [https://miau.my-x.hu/myx-free/index.php3?\_filterText2=\*konzisztencia](https://miau.my-x.hu/myx-free/index.php3?_filterText2=*konzisztencia) (ahol a függvény-szimmetriák, modell-szimmetriák, modell-láncok, stb. Végtelen komplexitása alapján írható elő az a szabály, mely megadja, mikor nem fecseghet egy végső megoldás, mely korlátozás lehet pl. adott egy/több objektumra vonatkozó, de lehet magára a modell vonatkozó is.

# Példa – Algoritmus-hatékonyság



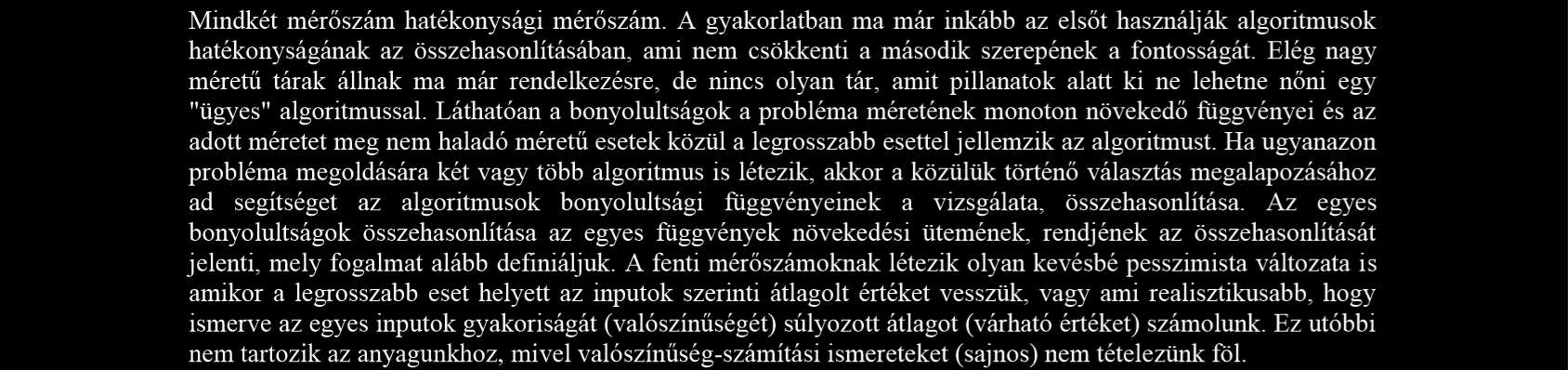
Ismét csak KNUTH-ra hivatkozva: tudás az, ami forráskódba átírható. Így a hatékonyság fogalma is akkor létező, s nem csak az intuitív/asszociatív folyamatok üzemanyaga, ha ennek levezetése automatizálható/algoritmizálható. Az emberi absztrakciók, mint pl. a hatékonyság fogalma általában véve többtényezős. Azok a fogalmak, melyek mérhetők (tömeg, hőmérséklet, stb.) önálló jelenségcsoportot képeznek. A hőérzet azonban olyan látszólag mérhetőséghez kötődő fogalom, mely még sem mérhető: „csak” algoritmikusan becsülhető (vö. <https://kiszamolo.com/hoerzet-index-kalkulator/>).

A fenti idézet alapján is többtényező alapján kell és lehet csak a hatékonyságról beszélni: vö.

* Lépésszám (ahol a lépés fogalma is egzakt módon tisztázandó és automatikusan mérhetővé teendő)
* Műveletszám (ahol a művelet fogalma is egzakt módon tisztázandó és automatikusan mérhetővé teendő)
* Tárigény (ahol a tárigény fogalma is egzakt módon tisztázandó és automatikusan mérhetővé teendő)
* …

A méretfüggőség problémája relativálással általában kezelhető: pl. egy nagyságrendre jutó átlagos számítási igény (vö. 4-jegyű vs. 100-jegyű szám alapján négyzetgyököt vonni). Azok a problémák, melyek a probléma méretével nem lineárisan arányosok, ennek megfelelően relativálandók.

A hatékonyságmérés alapját adó konkrét számhalmazok (inputok) mássága is relativálásra alapot adó jelenségcsokor.



(ahol a mindkét mérőszám nem más, mint időbonyolultság, tárigény-bonyolultság)

Ha nagyon szigorúan és/vagy idealista módon tekintünk az algoritmusok értékelésének, ill. a hatékonyság becslésének kérdésére, akkor a fenti idézet úgy értelmezendő, hogy addig nem illik elkezdeni semmit fejleszteni, amíg ennek minősítéséről nincs világos procedúra meghatározva (vö. BOSTROM - <https://www.google.com/search?q=bostrom+site%3Amiau.my-x.hu>). A mesterséges intelligencia fejlesztések alapja mindenkor a JÓ<JOBB<LEGJOBB fogalmának automatizált értelmezni tudása. A KNUTH-i elv értelmében: algoritmus-hatékonyságról beszélni csak algoritmikus alapon szabad. Így az első algoritmus-verzióval legkésőbb együtt ennek minden hatékonysági mérőszáma rendelkezésre kell, hogy álljon. Ha ezen előkészítő szint adott, akkor az első algoritmusnál JOBBAT készíteni már nem csak emberi intuícióval, hanem JOBB-kereső algoritmussal is lehet, melyet minden különösebb gond nélkül lehet mesterséges intelligenciának tekinteni (lévén ez az emberi intuíciót igénylő emberi aktivitás kiváltása, vagyis emberi képesség számítógépnek való átadása).

Hatékonysági szempont lehet akár a kezelendő komplexitások szintjeinek, arányainak felmérése is: pl. <https://miau.my-x.hu/miau/185/occams_razor_finetuned.doc> (az unatkozó kannibálok problémájának alternatív értelmezései és ezek rangsorolása).

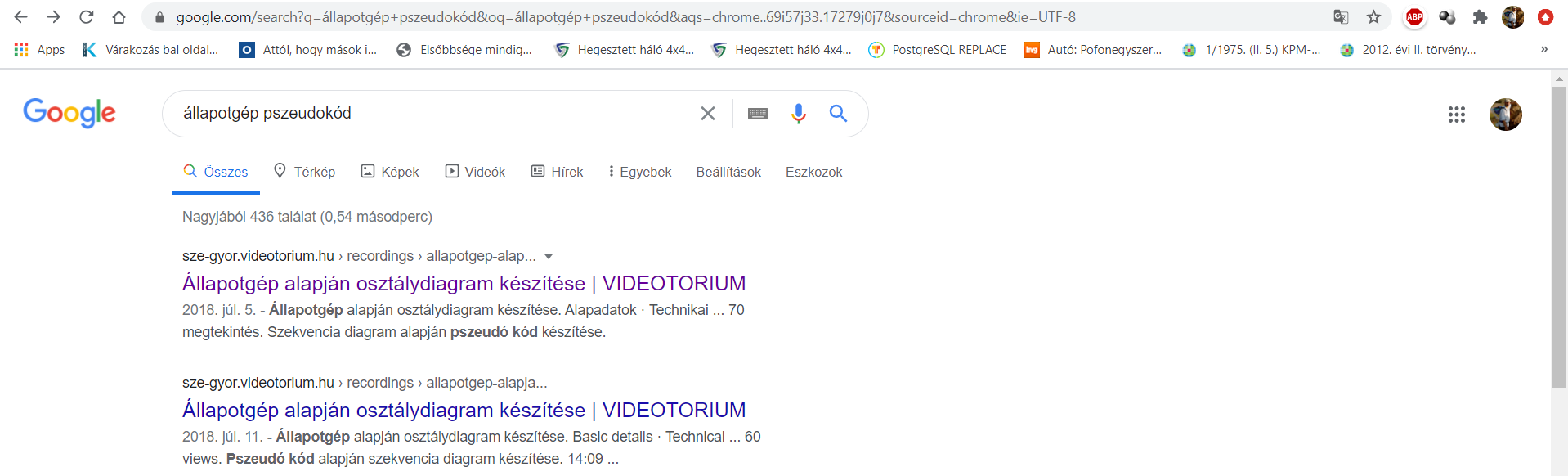
# Példa – Excel, mint szabványos pszeudo-nyelv

Itt kell megemlíteni a vallásháborúkhoz hasonló helyzetet, vagyis azt, hogy a fejlesztők/felhasználók egy része preferálja az Excel-alapú demo-megoldások létét, mások elvetik az Excel-alapúságot. Itt és most nem az a lényeg, kinek lehet esetleg egyáltalán igaza, hanem csak az, hogy az Excel-alapúság létezik, vagyis teszt-esetek, univerzális, de nem klasszikus programszerű megoldások Excel-ben is kialakíthatók, mintegy a klasszikus programozás megrendelését támogató segédletként, ahol ez a segédlet akár a részletes specifikációt is jelentheti, amennyiben az Excel demo megjelenése és funkcionalitása annyiban teljeskörű, hogy lokálisan (méretkorlátokkal, gyorsaságkorlátokkal, stb.), de elvégez mindent, amit el kell végeznie, megjelenít mindent, amit meg kell jelenítenie.

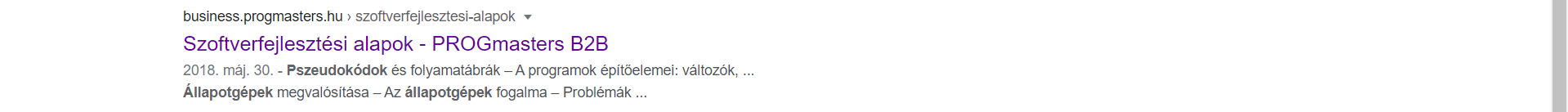
Így vetődik fel a kérdés, vajon ki és miért akarja, nem akarja az Excel-alapú megrendelést egy fajta univerzális, mindenkor teljesen szabványos pszeudo-nyelvnek, pszeudo-kódnak tekinteni?

Az Excel-ben minden lépés (ideális esetben) tételesen nyomon követhető, az egyes részletek egymástól való függősége, egymásból való következése tételesen értelmezhető, s mivel az Excel (verziónként kisebb-nagyobb eltérésekkel), de mindenki számára azonos keretet jelent, így az Excel-ben átadott információk értelmezhetősége is szabványosnak tekinthető. Ennek a szabványosságnak az operatív mércéje triviálisan az kellene, hogy legyen, hogy egy-egy Excel-demo alapján tetszőleges nyelven forráskód lenne képes keletkezni algoritmus alapján.

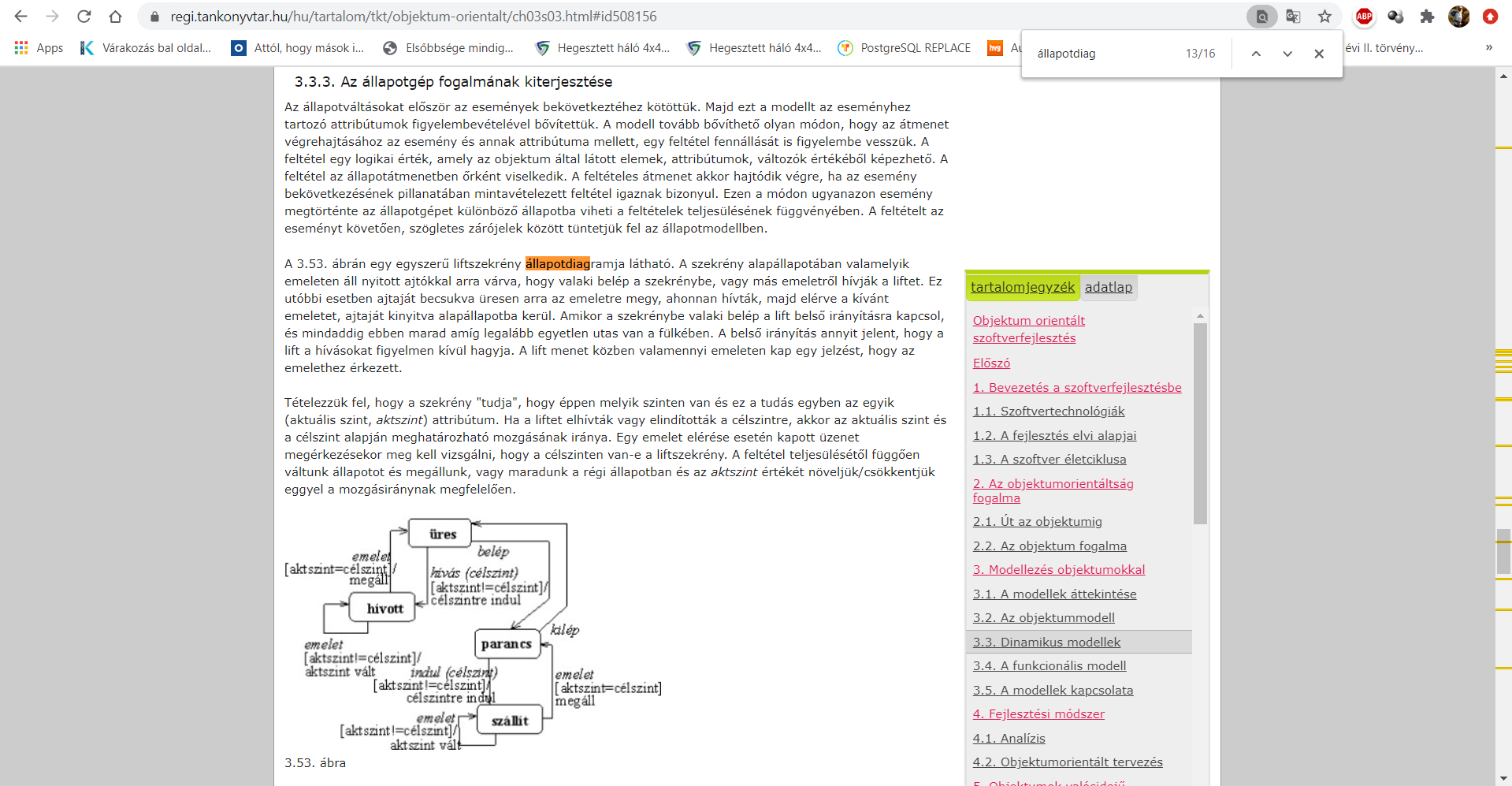
# Példa – Állapotgépek vs. pszeudo-kódok



…

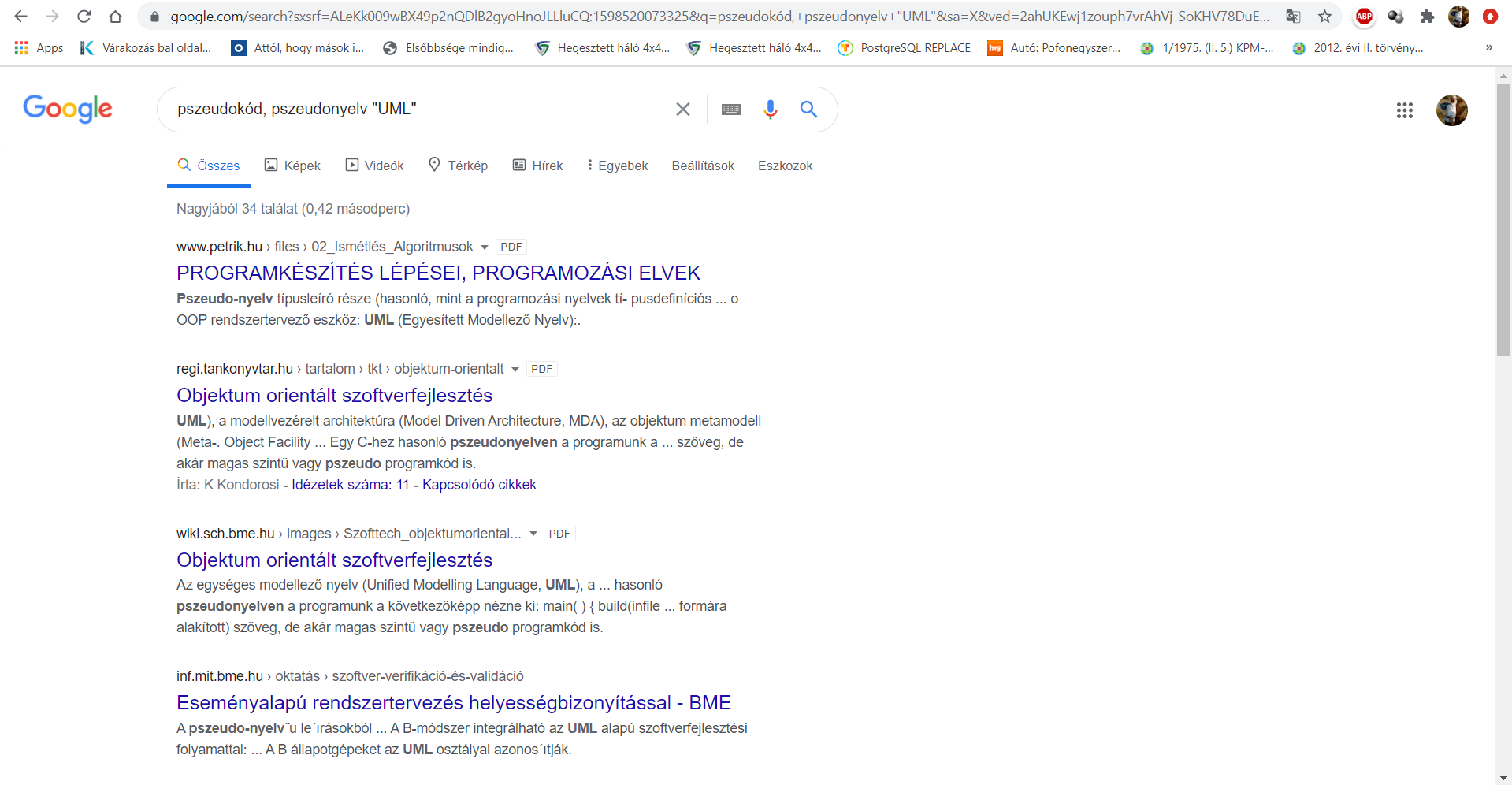


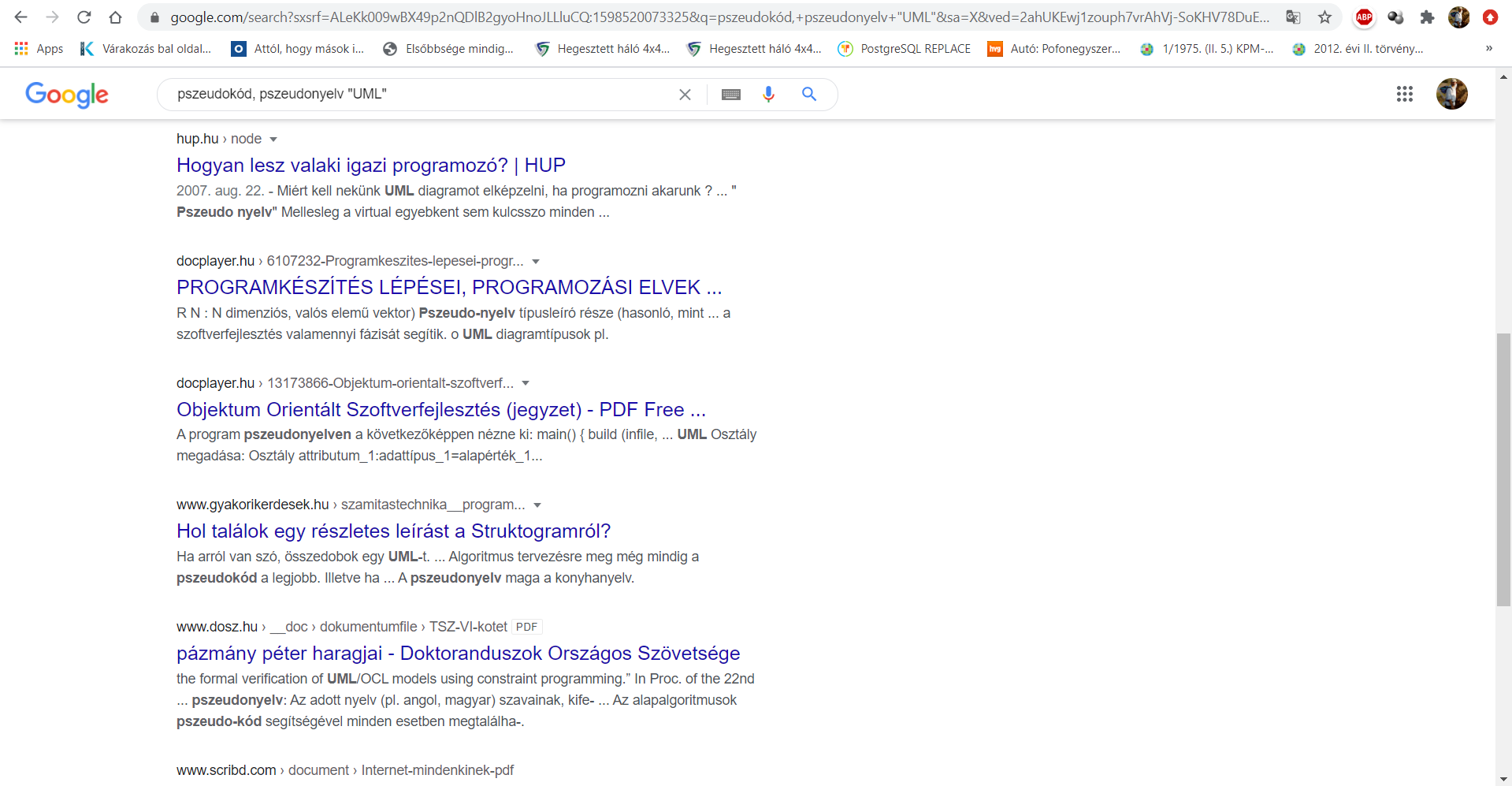
Az állapotgép és a pszeudo-nyelv közötti potenciális hidat az állapotdiagramok/kommunikációs diagramok jelenthetik (vö. <https://regi.tankonyvtar.hu/hu/tartalom/tkt/objektum-orientalt/ch03.html>):



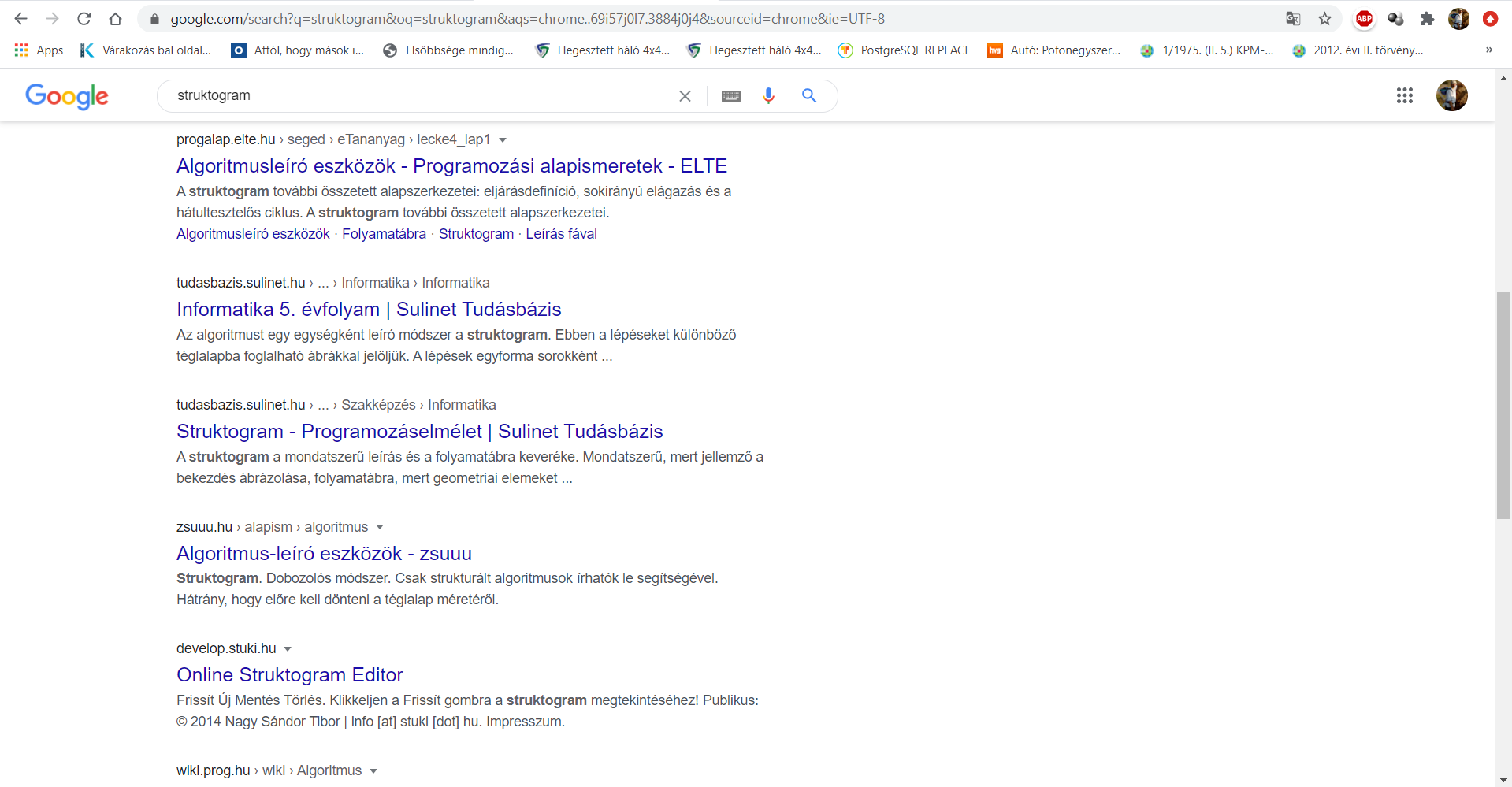
Az idézett háttéranyagok alapján az állapotgép funkcionalitása nem mutat lényeges eltérést a pszeudo-nyelvekkel kapcsolatos elvárásoktól, potenciáloktól.

# Példa – UML vs. pszeudo-kód





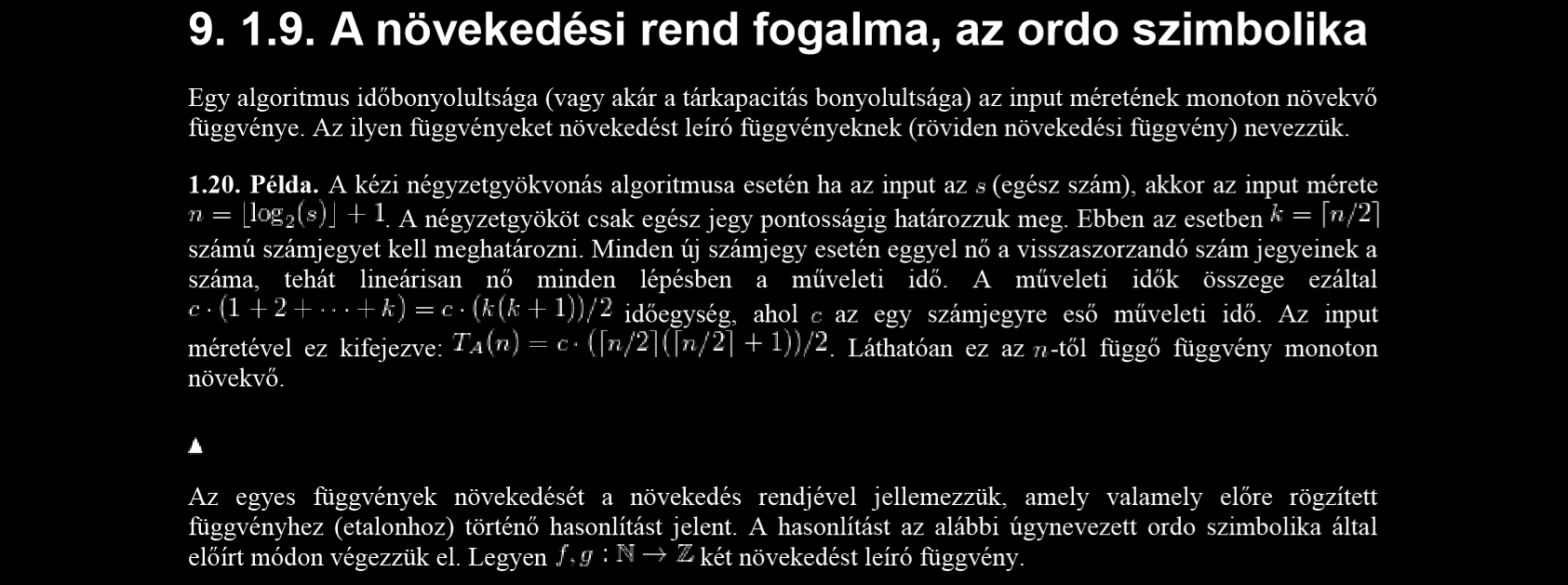
Amint az az autodidakta tanulást, problémamegoldást érzékeltetni kívánó Google-találati listák is jelzik, egy-egy fogalompár logikai kapcsolatáról (ha már az online keresés ingyenes rétegeiben a NEAR-operátor, mely képes lenne megadni a szavak közötti távolságot) nem minden esetben áll rendelkezésre) szimplán a kivonatos találati lista szövegtöredékei alapján is lehet képet alkotni. Ennek alapján az UML és a pszeudo-nyelv/kód közötti kapcsolat is kellően szorosnak vélelmezhető. Az elmúlt néhány példa kulcsszavai vélelmezhetően még tovább is bővíthetők (vö. struktogram). Így ezen kulcsszavak logikai kapcsolatának nem ártana valamiféle teljességre törekvő állapotát elérni, közelíteni.



Ennek a közelítésnek egy lehetséges módja a BPROF-képzés kapcsán tervezett komplex wiki-szócikk-sorozat, ahol az ontológiainak nevezett fejezetben (vö. <https://miau.my-x.hu/mediawiki/index.php/Szak%C3%A9rt%C5%91i_rendszer>) az „ez-egy“, a “van-neki”, ill. az asszociatív és minden egyéb lehetséges kapcsolat (vö. modus tollens = <https://en.wikipedia.org/wiki/Modus_tollens>, modus ponens = <https://en.wikipedia.org/wiki/Modus_ponens>, stb.) olyan kulcsszó-kapcsolatok tárhatók fel rendszerszerűen, melyeknek végső soron egymást erősítő (konzisztens módon) kell azt a képet kialakítani, mely leírja, mely kifejezés melyiknek része, vagy befogadója, ill. valamilyen mértékben, de nem teljeskörűen egymással kapcsolódnak, közös halmazt/halmazokat alkotva. Sajnos a tapasztalatok alapján ilyen konzisztens rendszerek nem egyszerűen alkothatók meg, vagy nem is léteznek adott kulcsszókészlet esetén. Visszagondolva a növények, mint objektumok és ezek attribútumainak egységes logikájú leírására, csak megerősödni látszik a KNUTH-i elv hiánya általában véve a leíró tudományok, a tudomány leíró aspektusai tekintetében, ami pedig a leíró tudás oktatásának, számonkérésének esetleges feleslegességét is magától értetődően felvetni engedi – bármilyen eretnek is egy ilyen felvetés a több évszázados oktatásmódszertani és tudománytörténeti hagyományok szerint. Itt emelve ki újra a példát: ha csak egy gomba ehetősége a kérdés, mint racionális döntési helyzetben elvárt egyetlen output, akkor semmilyen rendszertani részletre nincs szüksége a döntéshozónak soha semmikor ezen konkrét alkalmazási helyzetek végtelen sorozatában…

Egy fajta zárógondolatként megfogalmazható, hogy olyan tesztkérdésekre, mint pl. minden folyamatábra struktogram-e?, vagy minden struktogram folyamatábra-e? – Alexa jelenleg és talán soha a jövőben nem tud majd választ adni. S ha ez így van, akkor ilyen tesztkérdést Hallgatóknak feltenni, vagy Hallgatóktól befogadni más Hallgatók tesztelésére pl. a szakmai etika határaiba ütközik. Más megközelítésben: önálló kutatási feladat lehetne, a szakirodalomban kulcsszónak tűnő kifejezések halmazelméleti kapcsolatainak minél automatikusabb feltárása a szómágia zavarainak csökkentésére, ill. az előbb-utóbb gondolat (szöveg) vezérelt programozás alapjaként…

# Példa – PHD>MSC>BSC>ECDL



Phd-szintű kihívás sejtet a következő kérdés: Vajon a Solver-alapú gyökvonás kapcsán is érvényes-e a fenti gondolatmenet?

MSC-szintű kihívásként értelmezhető az időbonyolultság és/vagy tárkapacitás-bonyolultság fogalmainak matematikai értelmezései – de BSC/BPROF-szintű kihívás bármely fogalom racionális értelmezésének (helyes szómágikus használatának) elsajátítása.

A korábbi példák kapcsán nem volt tételesen részletezve, mely fókuszok milyen szintűnek becsülhetők. Az alábbiakban ennek pótlása következik példaértékűen (nem teljeskörűen):

* A Solver-alapúság már lehetne része az ECDL-nek, bármely tárgyként oktatott ismeretcsomagnak bármely szinten, hiszen a Solver-alapúság csak annyit mond ki, hogy minden feladat minden rendelkezésre álló segédeszköz felhasználása mellett oldandó meg – ahol egy másik ember nem minősül segédeszköznek értelemszerűen.
* Az ember-gép-szimbiózisra törekvés is ECDL-szinttől érvényesítendő elv, vagyis: amennyiben adott Hallgatói csoport/Hallgató nem rendelkezik a Solver-alapúságra utaló jelekkel validációs/PLA-dokumentációjában (részletes szakmai életrajzában), úgy ennek pótlása a következő (pl. BPROF/BSC) szint feladat.
* A KNUTH-i elv, ezen belül is a rendszertanokkal való együttműködni tudás is ECDL-szintű, s ennek hiánya esetén a pótlás mihamarabb meg kell, hogy történjen.
* A pszedu-nyelvek (ill. a funkcionálisan ekként értelmezhető jelenségek mindegyike: vö. folyamatábra, struktogram, konvenció-variánsok, UML, stb.) egységes rendszertani értelmezése ismét csak ECDL-szintet érint, vagyis már a magyar nyelv oktatása kapcsán a szövegértés objektivitásának növelését célzó törekvések említésével együtt kellett volna ezeket a problémákat is kezelni. S ennek elmaradása esetén a hiánypótlás az első, a hiányt feltáró eset után azonnal esedékes.
* Az intuíció szerepének fókuszba állítása is középiskolai feladat (lehetett volna).
* A hatékonyság fogalmának matematizálása a Solver-alapúság egyik középiskolai (ECDL-szintű) példája (lehetett volna).
* Az Excel-alapúság, mint a gondolkodás eredményének objektivizálási törekvése is ECDL-szintnél több IT tudást nem igényel(t volna).

Mint látható, a II. rész fókuszában olyan jelenségek állnak alapvetően, melyekre jelenleg az ECDL-szint értelmezése kapcsán lehetőség lenne (mindig is lehetőség lett volna), s hosszabb távon ezen gondolkodási mintázatok át kellene, hogy hassák először a pedagógus-továbbképzés rendszerét, majd innentől a környezetvédelemmel és/vagy a néptánccal együtt a bölcsődétől/óvodától induló mindennemű akkreditált oktatási elveket és gyakorlatokat. Ennek oka egyszerű: minden potenciális pedagógus képes, ha ő maga is így van szocializálva, folyamatosan rámutatni azon valóságelemekre, melyek automatizálása hosszabb távon elkerülhetetlen. S teheti ezt pl. a magát humán beállítottságúnak besoroló pedagógus anélkül, hogy tudnia illene, miként is kerül majd pl. az óvodában a zenei tehetség hangfelvételek alapján robotok által értelmezésre, ill. a gyógypedagógiában miként javasol a robot-logopédus adott beszéd-mintázatok alapján javító gyakorlatokat adott alany számára…

Az MI-alapúság tehát ugyanúgy át kell, hogy hassa a társadalmat, mint a környezet-érzékenyítés, a pénzügyi tudatosság, a kulturális gyökerek tisztelete…

További részletek: <https://miau.my-x.hu/bprof>