



**EÖTVÖS LORÁND TUDOMÁNYEGYETEM**

**INFORMATIKAI KAR**

**ALGORITMUSOK ÉS ALKALMAZÁSAIK TANSZÉK**

---

# Hordozható Struktogram

## Szerkesztő és Futtató környezet

témavezető:

**Veszprémi Anna**

mestertanár

szerző:

**Órcsik Antal**

programozó informatikus BSc

Budapest, 2016

## Felhasználói dokumentáció






A STRUKI egy struktogram szerkesztő és futtató környezet, mellyel összetett algoritmusok modellezhetők, és azok működése ellenőrizhető, szemléltethető. A programmal az ELTE Programtervező Informatikus BSc szak első két félévében bemutatott programozási tételek többsége létrehozható és futtatható is. A generált struktogram ábra egyszerűen kimenthető például dokumentációban történő felhasználás céljából.

### A program felépítése

A program egy HTML, CSS és JavaScript nyelven írt kliens oldali „web” alkalmazás, viszont mivel minden szükséges erőforrása megtalálható a csomagban, internet kapcsolat nélkül futtatható.

### A futtatáshoz szükséges környezet

A futtatásához csak egy „modern” böngésző program szükséges, melyben engedélyezve van a JavaScript kódok futtatása és a CSS stíluslapok betöltése. A programot az alábbi böngészőkkel és operációs rendszerekkel teszteltem. Amelyik verziónál nincs megjegyzés, ott a program az elvártak megfelelően működött:

	 Internet Explorer	 Edge	 Chrome**	 Firefox	 Safari
Windows 7	11.0*	–	50.0	38.8	–
Window 10	11.20*	25*	50.0	46.0	–
Mac OS X 10.11	–	–	50.0	45.1	9.1***

\* A file:/// protokollal történő futtatás esetén a lokális tárhely nem támogatott.

\*\* Az optimális felhasználói élmény érdekében a Chrome böngésző használatát javaslom.

\*\*\* Exportáláskor a fájlok új ablakban nyílnak meg, nem a letöltés párbeszéd panelen.

A beállítások és a dokumentumok automatikus mentéséhez szükséges, hogy a program hozzáférjenek a böngésző lokális tárhelyéhez, viszont ennek hiányában is használható a program.

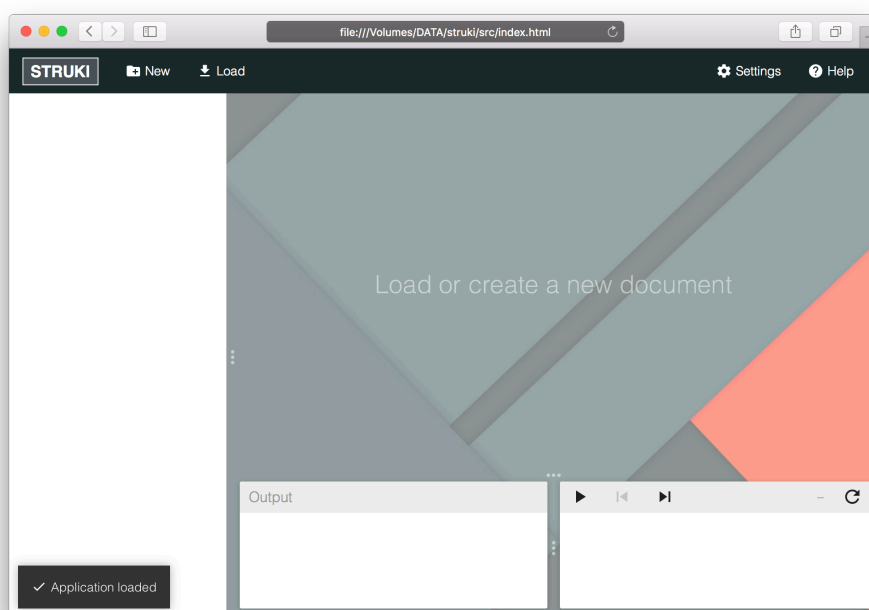
## A csomag tartalma

A megosztásra szánt, épített változat csak a legszükségesebb állományokat tartalmazza, összefűzve. Ebben a formában a program nem alkalmas arra, hogy azon javításokat vagy fejlesztéseket lehessen végrehajtani, ahhoz a forrás állományokra és egy megfelelően előkészített fejlesztői környezetre van szükség.

A futtatáshoz a csomag minden elemére szükség van, a mappák vagy állományok nevének vagy tartalmának megváltoztatása, törlése a programot használhatatlanná teheti. Ez alól kivétel az *examples* mappa, amiben példa állományok találhatók, melyeknek betöltése és tanulmányozása segítheti a program használatát, és a struktogram szerkesztés, az algoritmikus gondolkodás jobb megértését.

## A program indítása

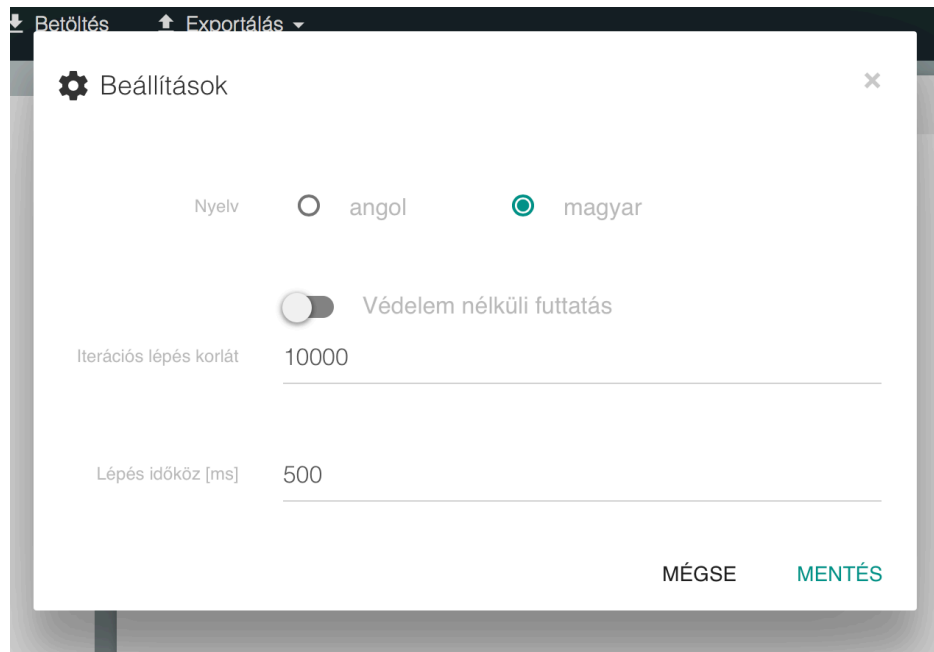
A programot nem kell telepíteni, (kitömörítés után, ha tömörítve érkezik) az index.html megnyitásával indítható, ami az alapértelmezett böngészőben fog megnyílni, de bármilyen böngészőben futtatható. Indítás után az alapértelmezett nézet jelenik meg (1. ábra). A bal alsó sarokban egy üzenet jelzi, hogy a program sikeresen betöltődött.



1. ábra – A program nyitó képernyője

## Beállítások

A program alapértelmezetten angol nyelven indul, de van magyar fordítás, amit az eszköztár fogaskerékkel jelölt menüpontjára kattintással előhívható beállítások panelen lehet módosítani.

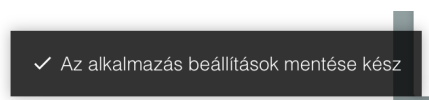


2. ábra – A beállítások panel

Ezen a panelen (2. ábra) állítható a védelem nélküli futtatás és az iterációs lépés korlát. A futtatás védelem alapértelmezetten be van kapcsolva, melynek hatására a struktogram futása megáll, ha az iteráció számláló eléri a meghatározott korlátot. Iterációs lépésnek számít például egy struktogram cellába írt kifejezés kiértékelése. Ennek elsődleges célja, hogy megvédje a futtató környezetet attól, hogy végtelen ciklus esetén hozzáférhetetlenné váljon.

Végezetül itt állítható még a lépés időköz, amivel milliszekundumban [ms] adható meg, hogy folyamatos futtatás esetén az egyes lépések között mennyi idő teljen el. Ez az érték szabadon állítható, de nagyon alacsony vagy nagyon magas értéket nincs értelme megadni. A javasolt alapbeállítás 500 ms, azaz fél másodperc. Ilyenkor még kényelmesen követhető szemel a futtatás, és belátható időn belül be is fejeződik egy összetettebb algoritmus is.

A beállításokat a mentés (save) gombra kattintással lehet menteni. Minden beállítás azonnal alkalmazására kerül, nem kell a programot újratölteni. A beállítások sikeres mentéséről a bal alsó sarokban egy fekete téglalapban megjelenő üzenetben küld értesítést a program (3. ábra).



3. ábra – értesítés

## A felület és annak átméretezése

A felület 5 részre tagolódik, felül az eszköztár, bal oldalt a szerkesztő, alul jobbra a nyomkövető, alul közepén a kimenet, a fennmaradó területen pedig a dokumentum választó és a struktogram ábrák jelennek meg (4. ábra).

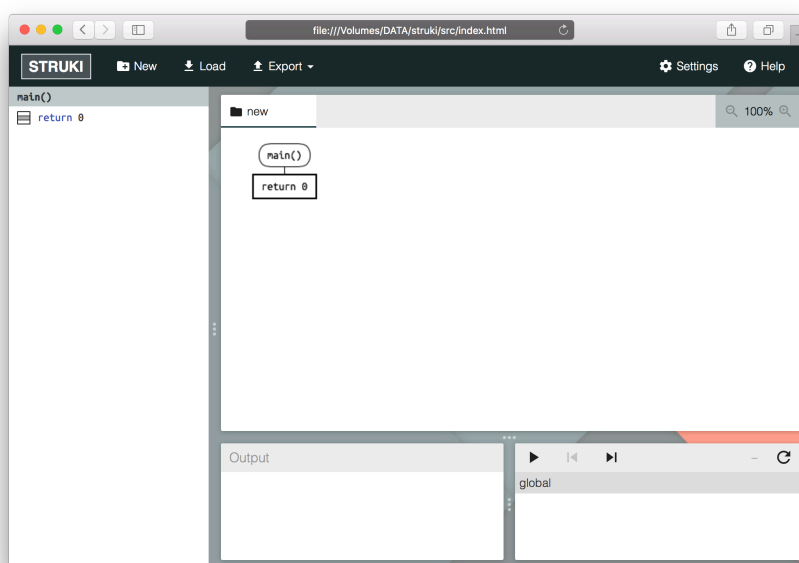
Az egyes komponensek egymáshoz képest százalékos arányban vannak elrendezve. Ezek az arányok a három pöttyel jelölt kis „fogantyúk” húzásával változtathatók meg. A böngésző ablak átméretezése során ezek az arányok megőrződnek, illetve mentésére kerülnek a lokális tárhelyre, vagyis két indítás között sem vesznek el.

## Dokumentum létrehozása

A *dokumentum* a program számára értelmezhető állomány, amiben több struktogram is lehet, de mindenképpen szerepel benne egy *main* nevű un. fő struktogram.

Új dokumentumot az eszköztár Új menüpontjára kattintással lehet létrehozni. Ekkor a középső területen megjelenik a dokumentum választó és grafikus komponens (4. ábra). A dokumentum választó egy fülecskékből álló vízszintes menüsor, amiben az összes megnyitott dokumentum szerepel a nevével. A grafikus komponensen pedig megjelent egy alap struktogram, aminek a neve `main()`, és szekvenciájában egy `return 0` kifejezést tartalmazó parancs található. Eközben megjelent a szerkesztő nézetén a dokumentum struktogramjának megfelelő szerkeszthető pszeudo kód lista, illetve az eszköztáron a Betöltés menüpont mellett az Exportálás.

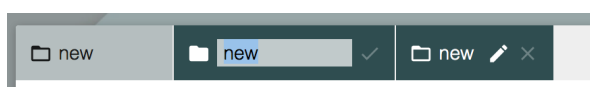
A dokumentum választó jobb oldalán látható két nagyító között egy 100% felirat. Ezzel lehet a grafikus nézeten megjelenő struktogramok méretet szabályozni. A bal oldali nagyítóval csökkenteni, a jobb oldalával növelni, 50% és 150% közötti intervallumban. Ez akkor jön jól, ha több nagy struktogram van a dokumentumban, melyeket kisebb méretben könnyebb átlátni.



4. ábra – A felület, egy új dokumentummal

## Dokumentum szerkesztése

A dokumentum választó listában a módosítani kívánt név fölé mozgatva az egeret megjelenik egy kis ceruza ikon, erre kattintva a név egy szerkesztő mezőben jelenik meg. Módosítás után az **Enter** gomb lenyomására, vagy a kis pipa alakú mentés ikonra kattintva menthető az új név. A szerkesztést az **Esc** billentyű lenyomásával lehet megszakítani (5. ábra).










5. ábra – A dokumentum nevének szerkesztése

## Dokumentum bezárása

A füleken található még egy kis „x” ikon is, erre kattintva bezárható az adott dokumentum. Bezárás után, a tőle jobbra eső, vagy ha ilyen nincs akkor balra eső nyitott dokumentum lesz aktív. Ha az összes dokumentumot bezártuk, akkor a program az induláskor látott alaphelyzetbe kerül.

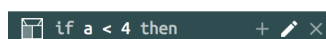
## Struktogram szerkesztése

A dokumentumban található struktogramok szerkesztése a bal oldali szerkesztő komponensen keresztül érhető el. Az egyes struktogramok egymás alatt sötétebb fejléccel helyezkednek el. A fejléc alatt pedig a vezérlési elemek egymás alatt. Ha egy elem tartalmaz más elemeket is, akkor azok bentebb kezdve jelennek meg. Minden sor elején látható az elem típusát jelző kis piktogram.

-  Struktogram
-  Parancs
-  Ciklus
-  Elágazás (csak a hozzáadás menüben jelenik meg)
-  Ág
  
-  Skip (ez csak akkor jelenik meg, ha egy szekvencia üres)
-  Deklaráció (a struktogram fejlécéhez tartozó lokális változók)

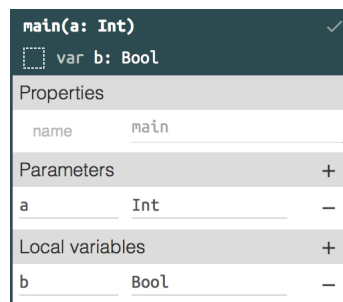
Az egyes sorok, a bennük tárolt kódrészlet szövegén kívül, tartalmazhatnak még kiegészítő szövegeket, mint például ciklus esetén a „while”, vagy elágazásnál az „if” szócska (6). Ezek a szerkesztőben megjelenő tartalomnak egyfajta pszeudokód jelleget kölcsönöznek, ami tovább segíti a megszerkesztett algoritmus értelmezését.

Itt is a már fentebb ismertetett kis ceruza ikonnal lehet szerkesztő módba kapcsolni egy elemet, ami az egér elem fölé mozgatasakor jelenik meg (6. ábra).



6. ábra – szintaktikai szócskák és az elem szerkesztő ikon

Az egyes elemek szerkesztő módja a rájuk jellemző adatok függvényében más és más. A struktogram szerkesztőjében egy szöveg mező található, amivel a nevet lehet módosítani (kivéve a main struktogramot, annak a neve nem módosítható). Található benne továbbá egy paraméterek és egy lokális változók űrlap fejléc, egy kis plusz jellel. A plusz megnyomására egy új paraméter vagy lokális változó mező pár jelenik meg, aminek az első tagjába a név a másodikba a típus kerül. Egy-egy ilyen sort a mínusz ikonnal lehet törölni (7. ábra).



7. ábra – struktogram elem szerkesztő mód

A típus értéke szabad szöveges paraméter, de futtatható struktogram csak az alábbi típusokat tartalmazhatja:

- `Int` egész szám
- `Float` lebegő pontos szám
- `Bool` logikai érték
- `String` szöveg
- `Array` tömb (az indexelés 0-tól indul)

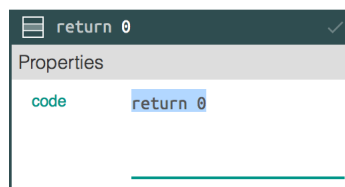
A tömbök (a program JavaScript alapjainak köszönhetően) bármilyen típusú elemet tartalmazhatnak, de deklarálhatók `Int*` típussal is, ha jelezni akarjuk, hogy egész számokat tartalmazó tömbről van szó. A program futtatáskor minden `*` végződésű típus nevet tömbként azonosít, és ennek megfelelően vizsgálja az egyes kifejezések típus követelményeit.



A mentéshez a kis pipa ikonra kell kattintani, a szerkesztés megszakítása pedig az **Esc** billentyű lenyomásával érhető el. A struktogram szerkesztő és az ábra a mentés után azonnal megjeleníti a módosításokat. A paraméterek a struktogram neve utáni zárójelben jelennek meg, a lokális változók pedig alatta **var** szócska és deklaráció piktogram jelzéssel.

## Parancs szerkesztése

A dokumentum jelenleg csak egy parancsot tartalmaz, amit a struktogramhoz hasonlóan a ceruza ikonra kattintva szerkeszthetünk. A parancs szerkesztő módja csak egy szöveg mezőt tartalmaz, amibe a kifejezést írhatjuk, majd a pipa ikonnal menthetjük a változtatást (8. ábra).



8. ábra – parancs elem szerkesztő mód

## Struktogram kifejezések

A parancs kód mezőjébe bármilyen szöveg írható, de ha futtatható dokumentumot akarunk létrehozni, akkor be kell tartani a futtatható kifejezésekre vonatkozó szintaktikai szabályokat. Egy futtatható kifejezés tartalmazhat szám, logikai, szöveg és tömb literálokat, változókat, az ezeket összekapcsoló operátorokat valamint függvény hívásokat. A **return** kulcsszóval pedig struktogram visszatérési értéke adható meg.

## Foglalt szavak

A fentebb már említett **return** kulcsszó foglalt szó, azaz csak a neki fenntartott szerepben használható. További foglalt szavak: **if**, **then**, **else**, **while**, **until**, **for**, **var**, **in** és az **\_** (alul vonás). Ezek többsége a korábban említett szintaktikai szócska, azaz nem része a fordítható kifejezéseknek, viszont ha engedélyezett lenne a használatuk változóként, akkor az átláthatatlanná tenné a szerkesztő nézetet.

## Literálok

A literálok az egyes érték típusok szöveges reprezentációi, mint például:

- semleges érték: `NIL`
- logikai érték: `I`, `H`
- egész szám: `0`, `1`, `-1`, `1000`
- lebegő pontos szám: `-1.5`, `0.0001`
- szöveg: `"pelda \"szöveg\" literálra"`
- tömb: `[]`, `[0,1,2]`, `["alma", "körte", 4]`

A szöveg literálban `\` (visszaper) karakternek kell megelőznie az `"` és a `\` karaktereket.

## Változók

Változó név lehet bármely legalább egy, `_` (alul vonást), az angol ABC kis és nagy betűit vagy számot tartalmazó karaktersorozat, de nem kezdődhet számmal. Tehát helyes változónevek például az `a`, `nev`, `a2`, `_1`, de helytelenek a `1nev` vagy a `név`.

Az `_` (alul vonás) önmagában foglalt szó, azaz nem lehet változóként deklarálni, viszont használható változóként. Ez a speciális változó az úgynevezett *nyelő*, aminek az értéke mindig semleges, és bármit értékül lehet neki adni. (Erre a többes értékadásnál még visszatérünk.)

## Operátorok

Minden típus esetében használható operátorok:

- `=`, `!=` egyenlő, nemegyenlő
- `:=` egyszerű, vagy többes értékadás

Egyszerű értékadásnál a bal oldalon változó, vagy egy tömb elem kell álljon, a jobb oldalon pedig egy megfelelő érték. Többes értékadás esetén mindkét oldalon, több elem szerepelhet, vesszővel elválasztva. Fontos, hogy a két oldalon az elemek száma meg kell egyezzen. Ha nincs szükség valamelyik visszaadott értékre, akkor ott a korábban ismertetett *nyelő* változót lehet használni, pl.: `x`, `y[0]`, `_ := 1, 2, 3`

Egész és lebegő pontos szám operátorok:

- `+`, `-` összeadás, kivonás (negáció)
- `*`, `/` szorzás, osztás
- `%`, `^` maradékos osztás, hatványozás
- `<`, `<=` kisebb, kisebb vagy egyenlő
- `>`, `>=` nagyobb, nagyobb vagy egyenlő
- `+=`, `-=` növelő-, és csökkentő értékadás

Logikai operátorok:

- `&`, `|` logikai és, logikai vagy
- `!` negáció

Szöveg és tömb operátorok:

- `+` összefűzés
- `a[n]` indexelés, az `a` változóban tárolt tömb vagy szöveg `n`. indexű eleme

Speciális operátorok:

- `..` intervallum  
A jobb és bal oldalon álló egész számokkal és az azok közötti egészekkel létrehoz egy tömböt, pl.: `1..3 = [1,2,3]`
- `,` vessző  
Elemek listája definiálható vele, ami tömb literálban, függvényhívás paraméter listájában vagy többes értékadásban szerepelhet, pl.: `print(1, 2)`
- `in` tartalmazás  
megállapítja, hogy a bal oldalon szereplő érték benne van-e a jobb oldalon szereplő tömbben, pl.: `1 in [0,1,2] = 1`
- `()` zárójelek  
zárójelezéssel csoportosíthatók az összetett kifejezések, így felülbíráható azok természetes preferenciája

## Függvényhívások

Kétféle függvény hívható a kifejezésekben, az egyik maga a struktogram, amit nevével és megfelelő paraméter listájával lehet meghívni, a másik a beépített függvények.

A dokumentumban létrehozott struktogramok mind elérhetőek lesznek bárhol. Ezek visszatérhetnek többes értékekkel is, amiket többes értékadásnál használhatunk.

A beépített függvényeket a keretrendszer biztosítja.

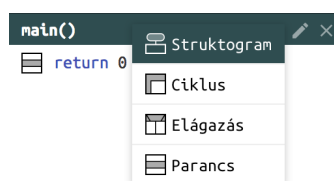
- `print(a, ...)` a paraméterül kapott érték listát a kimenetre írja
- `size(A)` visszaadja az `A` tömb vagy szöveg hosszát

A fentieken kívül elérhető a JavaScript Math objektumának minden függvénye és konstansa. Azaz, például a Math objektumban van egy `Math.cos(d)` függvény, és egy `Math.PI` konstans, ezek a `Math.` előtag elhagyásával struktogram kifejezésekben is használhatóak, pl.: `print(cos(PI)) = -1`

## Vezérlési szerkezetek

A szerkesztés ikon mellett bal oldalon (struktogram, ciklus és ág esetén) egy plusz jellel jelölt hozzáadás ikon is található. Erre kattintva egy menü jelenik meg, benne a hozzáadható elemek nevével és a már korábban látott piktogramokkal.

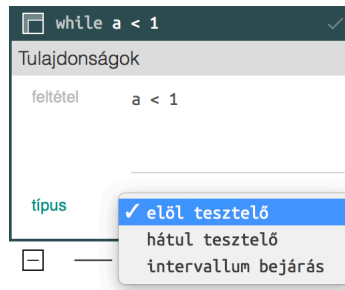
Struktogram esetén ez a menü (9. ábra) tartalmazza a struktogramot, ami a dokumentumhoz ad egy újabbat, illetve a ciklus, az elágazás és a parancs elemeket, melyeket az aktuális struktogram szekvenciájának végéhez fűz hozzá. A hozzáadott elem szerkesztő módban jelenik meg.



9. ábra – a struktogram hozzáadás menüje

## Ciklus szerkesztése

A ciklus két beállítható tulajdonságot tartalmaz, a feltételt és a típust. A típust egy választó űrlap elem segítségével lehet megadni, aminek három lehetséges beállítása az elől tesztelő, a hátul tesztelő és az intervallum bejárás (10. ábra).

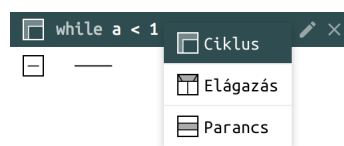


10. ábra – a ciklus szerkesztő módja

Az elől tesztelő és a hátul tesztelő típusú ciklus feltétel mezőjébe egy, a parancsnál már ismertetett szintaktikájú, logikai értékű kifejezést kell írni, pl.: `a < 1`

Az intervallum bejárás típusú ciklus esetén egy speciális intervallum feltételt kell megadni. Ennek formája egy korábban deklarált változó neve, amit az `in` (tartalmazás) vagy az `:=` (értékadás) operator, illetve egy tömb értékű kifejezés követ. A ciklus futása során a tömb elemeit ciklusonként egyesével beleírja a program bal oldali változóba. Ilyen speciális feltétel pl.: `b in ["alma", "körte", "barack"]`

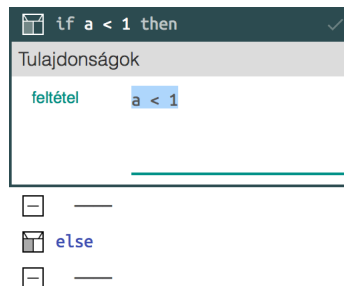
A ciklust a többi elemhez hasonlóan a kis pipa ikonra kattintva lehet menteni, illetve az **Esc** billentyűvel szakítható meg a szerkesztés. Hozzáadás menüjében a ciklus, elágazás és parancs elemek szerepelnek (11. ábra).



11. ábra – a ciklus hozzáadás menüje

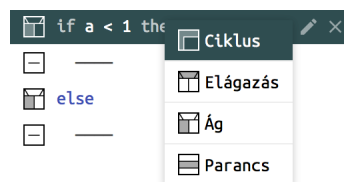
## Elágazás szerkesztése

Egy új elágazás mindig legalább két ágat tartalmaz, egy feltételhez kötött és egy „egyébként” ágat. Az elágazás maga csak ágak listája, amihez további ágakat a hozzáadás menüből lehet az adni.



12. ábra – az ág szerkesztő módja

A feltételhez kötött ágak szerkeszthetők (12. ábra) és ezek egyetlen tulajdonsága a feltétel mező, amibe az elől- illetve hátul tesztelő ciklusokhoz hasonlóan egy logikai értékű kifejezést kell írni. Az elágazás hozzáadás menüje a ciklus, elágazás, ág és parancs elemeket tartalmazza (13. ábra).



13. ábra – az ág hozzáadás menüje

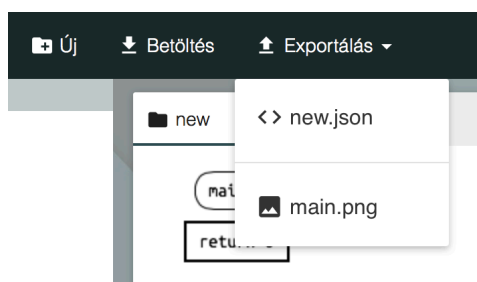
## Struktogram elemek törlése

A hozzáadás és szerkesztés ikonok melletti kis x segítségével lehet egy vezérlési elemet törölni. Kattintás után a program rákérdez, hogy valóban el szeretnénk-e távolítani az elemet, jóváhagyás esetén pedig törli a dokumentumból. A main struktogram törlés ikonja a dokumentumot zárja be.

☞ **Vigyázat!** Összetett elem esetén a tartalmazott elemek is törlésre kerülnek!

## Exportálás és betöltés

A fentebb leírtak alapján már szerkeszthető tetszőleges összetett struktogram. Hogy ezt később ne kelljen újra megszerkeszteni, a kész struktogram kiexportálható egy JSON formátumú állományként. Ehhez az eszköztár *Exportálás* pontjára kell kattintani a lenyíló menüben pedig a .json kiterjesztésű linke. Ennek hatására a böngésző letöltés párbeszéd ablaka nyílik meg, ahol kiválasztható hova kerüljön a állomány. A struktogram ábrákat a megfelelő .png kiterjesztésű linkekre kattintva lehet exportálni (14. ábra).



14. ábra – Exportálás menü

**Jegyzet:** A Safari böngésző sajnos nem támogatja ezt a fajta „letöltés” funkciót, ott egy új ablakban nyílik meg az állomány, amit a böngésző *Fájl* menüjének *Mentés...* pontjával lehet menteni.

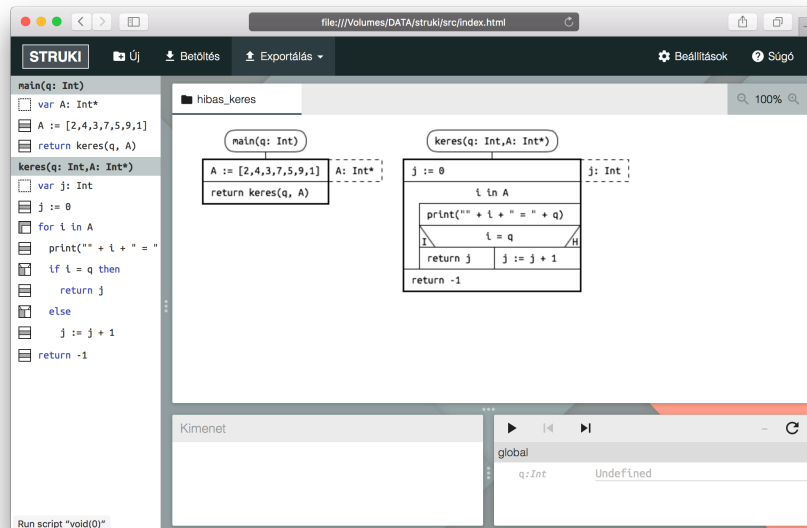
A kimentett JSON dokumentum visszatöltése is hasonlóan egyszerű feladat. Az eszköztár *Betöltés* pontjára kattintva megnyílik a böngésző program fájl feltöltés párbeszéd ablaka, ahol kiválasztható a felhasználó gépén található állomány. Sőt, úgy is megnyitható egy dokumentum, hogy egyszerűen rá húzzuk a felületre a böngésző ablakon kívülről.

## Automatikus mentés

A program automatikus mentéseket készít a megnyitott dokumentumokról a böngésző lokális tárhelyére, ha elérhető ez a szolgáltatás. Ezek a mentések a program újbóli nyitáskor helyreállítják az utolsó mentett állapotot. Mivel minden módosítás után készül automatikus mentés, így a böngésző vagy a számítógép váratlan hibája esetén sem vesz el a ki nem exportált munka. (Ha sérül a lokális tárhely tartalma, akkor természetesen az automatikus mentések elveszhetnek.)

## Futtatás

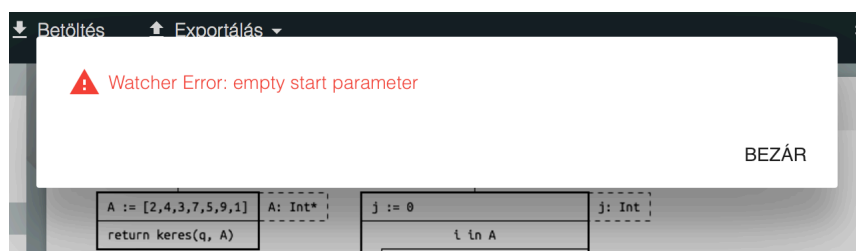
A futtatás szemléltetéséhez indítsuk el a programot és töltsük be a csomagban található *examples* mappából a *hibas\_keres.json* állományt (15. ábra).



15. ábra – *hibas\_keres.json* betöltve

Ez a kis algoritmus a paraméterül kapott számot próbálja megkeresni egy előre definiált tömbben. Ha megtalálta visszaadja a keresett szám indexét, ha nem akkor pedig -1-et.

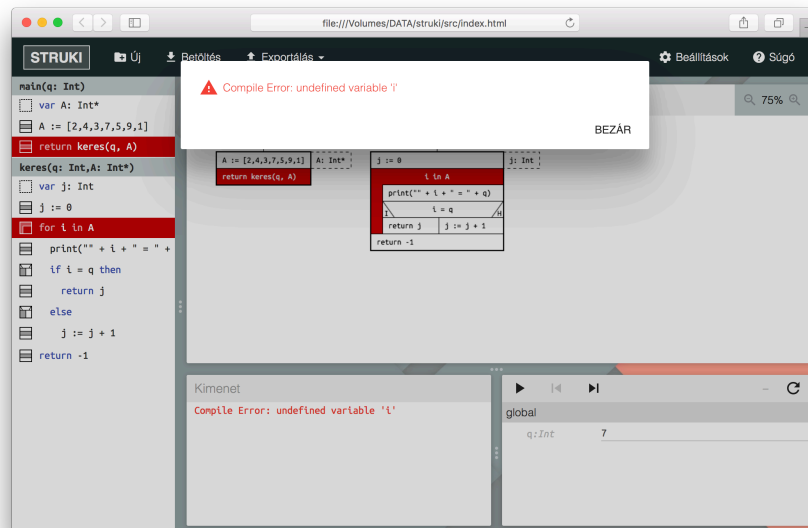
A képernyő jobb alsó részében a nyomon követés komponensben látható, hogy a globális környezet egy futtatási paramétert vár (`a: Int`). Ha úgy kattintunk a futtatás, vagy a léptetés gombra, hogy ezt nem töltjük ki, akkor a program hiányzó indítási paraméter miatt hibát fog jelezni és a futtatás nem indul el (16. ábra).



16. ábra – hiányzó indítási paraméter hiba



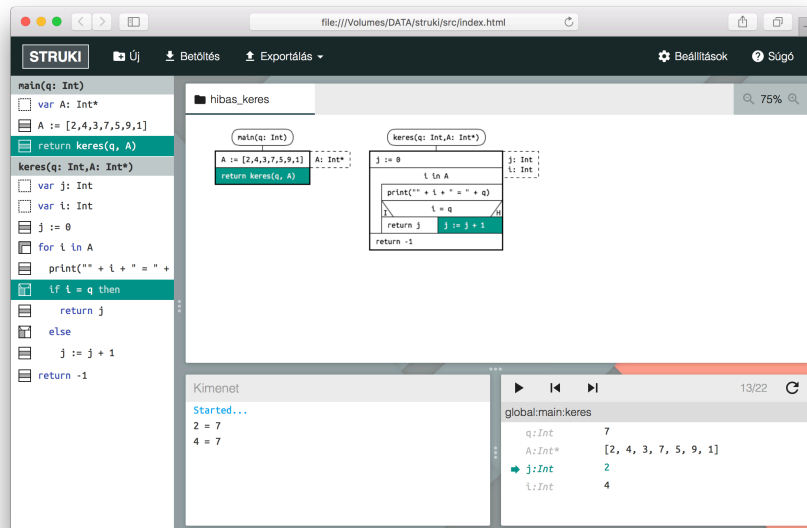
Adjuk meg a 7-es számot és próbáljuk elindítani a dokumentumot. A futtatás továbbra sem fog elindulni, és az alábbi hiba üzenetet fogja a program kiírni, valamint megjelöli a hibás kifejezést a szerkesztőben és az ábrán is (17. ábra).



17. ábra – fordítási hiba

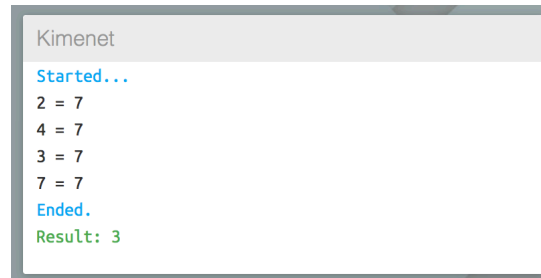
A probléma, hogy az intervallum bejárás ciklusban használt futó változó nem lett a struktogram fejlécében deklarálva. Ezt könnyen javíthatjuk, szerkesszük a `keres` struktogramot, adjunk hozzá egy `Int` típusú `i` lokális változót, és mentsük a kis pipára kattintással. Látható, hogy alaphelyzetbe került a felület. Ha most kattintunk a futtatásra, már el fog indulni a futtatás.

Az ábrán és a szerkesztőben is nyomon követhetjük, hol tart a program kiértékelése, valamint a kimeneten is látszik, hogy az egyes összehasonlításokat sorban kiírja a program (18. ábra). A futás bármikor megállítható a szünet ikonra kattintva. A nyomon követő komponensben, a vezérlő gombok alatt áttekinthető a pillanatnyilag futó struktogram környezete, az abban definiált változókkal és azok értékeivel. Az aktuális iterációban történt változást zöld nyilak jelzik. Ilyenkor elindítható, előre és hátra léptethető a futtatás, de meg is szakítható az alaphelyzetbe ikonnal állítással.



18. ábra – futtatás közben

Sikeres futás végen a kimeneten látható, hogy a 7-est a 3. indexű helyen találta meg az algoritmus (19. ábra).



19. ábra – sikeres futás kimenete

Érdemes átnézni az *examples* mappában található további példa dokumentumokat is.

Remélem a program hasznos segítség lesz a tanulmányok vagy az oktatás során!