SZINTAKTIKA

```
<solution>       ::=  {<oc-list>}
                      <sol-def>
<oc-list>        ::=  <openlist>
                      <closedlist>
<openlist>       ::=  "(:openlist " <integer> " " {<listelement>} ")"
<closedlist>     ::=  "(:closedlist " <integer> " " {<listelement>} ")"
<listelement>    ::=  "(" <integer> " " <string> ")"
<sol-def>        ::=  "(:sol " <integer> " " {<string>} ")"
```

SZEMANTIKA

A <solution>-ben szereplő <oc-list>-listában lévő első <oc-list>-ben az <openlist>-ben és <closedlist>-ben szereplő <integer>-ek értéke zérus, míg az utána következő <oc-list>-ekben lévő <openlist>-ekben és <closedlist>-ekben szereplő <integer>-ek értéke egyenként növekszik, és ugyancsak megegyezik.

Az <oc-list>-listában lévő első <oc-list>-ben az <openlist> <listelement>-listája egyetlen elemű (a kezdőváros, és a hozzá tartozó összköltség), miközben a <closedlist> <listelement>-listája kezdetben üres.

A <listelement>-ekben szereplő <integer>-ek értéke a hozzájuk tartozó <string>-ekben megadott városok – kiindulási és célvároshoz viszonyított – összköltsége (f=g+h).

Az <openlist>-ekben és <closedlist>-ekben szereplő <listelement>-listák összköltség szerinti növekvő sorrendben vannak felsorolva. Mindkét lista ismétlődés-mentes (azaz egy-egy város legfeljebb csak egyszer szerepelhet bennük).

A <sol-def>-ben szereplő <string>-lista a keresési probléma megoldása (a kiindulási városból a célvárosba vezető legrövidebb út): első eleme a kiindulási város, az utolsó eleme pedig a célváros. Az <integer> a <string>-lista által megadott megoldási út útköltsége.

INFORMÁLIS LEÍRÁS

A programok kimenete tehát egy szövegfájl (*output.txt* – ugyanabba a könyvtárba, ahol a *JAR* található), aminek adott a szintaxisa, és a szemantikája (lásd. fentebb). E szövegfájl 2 részből tevődik össze, amely részek adott sorrendben követik egymást. Az első rész **(1)** egy lista, amely elemei lista-párok (egy OpenList, és hozzá egy ClosedList). A második rész **(2)** pedig lényegében maga a megoldás, egy lista, városok egy sora,

avagy az optimális út a kiinduló városból a célvárosba, és ennek teljes útköltsége. Balról jobbra haladva tehát e lista első eleme a kiinduló város, utolsó eleme pedig a célváros.

Az első részt az érthetőség érdekében fejtsük ki valamivel bővebben. Itt tehát lényegében egy 2-szintű egymásba ágyazottságról van szó: listák listája. Miért van szükség erre a viszonylag összetett adatstruktúrára? Válasz: azért, hogy nyomon lehessen követni a program által megvalósított A* algoritmus menetét. A program ugyanis az A* algoritmus végrehajtása során egy kezdővárosból kiindulva lépésről-lépésre 2 listát bővít: egy úgynevezett OpenList-et (amit hullámfrontnak is szokás nevezni), amelynek elemei a következőkben kifejthető (Open) városok; és egy ClosedList listát, amely lényegében a már kifejtett (Closed) városokat sorolja fel.

Kezdetben az OpenList listának csak egyetlen eleme van: a kiinduló város. A ClosedList lista kezdetben üres. A városokhoz tartozik még egy-egy *f(n)* költség is, amely az adott városba adott útszakaszon történő eljutás útköltségének, és a célig még hátralevő, becsült útköltségnek az összege: az *összköltség*. A kiinduló város esetében ez nyilván (0 + a heurisztika értéke a kiinduló városban). A célvárosban viszont a heurisztika értéke zérus, így az összköltség ekvivalens az útköltséggel.

Mind az OpenList, mind pedig a ClosedList listában szereplő városok mellett ott van tehát egy-egy `f(n)` költség-érték. Miért van erre szükség? A válasz egyszerű: egyrészt tudni szeretnénk, hogy adott lépésben az OpenList-ből mely város(ok) jöhet(nek) szóba a kifejtésnél. Az A* algoritmus szerint, mohó módon mindig a legkisebb összeköltésűek közül kell választanunk. Előfordulhat azonban, hogy egy-egy városba többféle úton is el lehet jutni. Tehát más-más városok kifejtésén keresztül is eljuthatunk egy-egy adott városba. Ekkor nyilván az OpenList-ben, illetve akár a ClosedList-ben többször is szerepelhetne az adott város, ámde esetleg különböző `f(n)` költséggel (lévén különböző utakon jutottunk el hozzá). Ilyet azonban mi most **NEM** engedünk meg. Amennyiben egy adott lépésben egy adott városhoz kisebb összköltséggel is sikerül eljutni, vagy egy ilyen várost fejtünk ki a későbbiekben, úgy a megfelelő listában (OpenList-ben, vagy éppen ClosedList-ben) a nagyobb értékűt le kell cserélnünk erre.[1]

Végső soron tehát így alakul ki az `<oc-list>`, ami listák listája. Vannak OpenList-ek és a ClosedList-ek, amik lényegében város-listák. Viszont OpenList-ből, és ClosedList-ből több is van. Éppen annyi, ahány lépést teszünk az A* keresés során. Ez is egy lista tehát – listák listája.

A web-oldalról letölthető példa-megoldás jól szemlélteti ezt a koncepciót.

[1] Ilyen eset egyébként például akkor fordulhat elő, ha az útköltségekre nem teljesül a háromszög-egyenlőtlenség.