EÖTVÖS LORÁND TUDOMÁNYEGYETEM

Informatikai Kar

Numerikus Analízis Tanszék

Poligonok szűkítése és bővítése

Témavezető:
Dr. Lócsi Levente
adjunktus

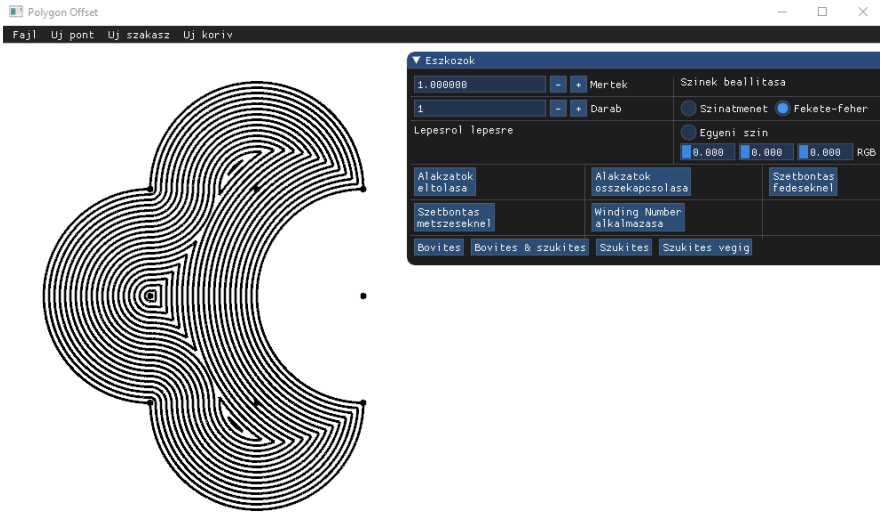
Készítette:
Oláh Dávid
programtervező informatikus BSc

Budapest, 2019

Tartalomjegyzék

1. Bevezetés

A szakdolgozatom egy olyan program elkészítéséről szól, amely képes két dimenzióban poligont tervezni grafikus felületen. A program implementál egy poligonszűkítő, illetve -bővítő algoritmust is, amivel a megtervezett poligont tetszőleges mértékben és darabszámban lehet szűkíteni vagy bővíteni. A szűkítés/bővítés jelentése a poligon összes élének eltolása egy adott irányba, adott mértékkel, majd ezen élekből egy új poligon képzése. Az alkalmazás képes megjeleníteni az algoritmus lépéseit és eredményét grafikusan.



2. Felhasználói dokumentáció

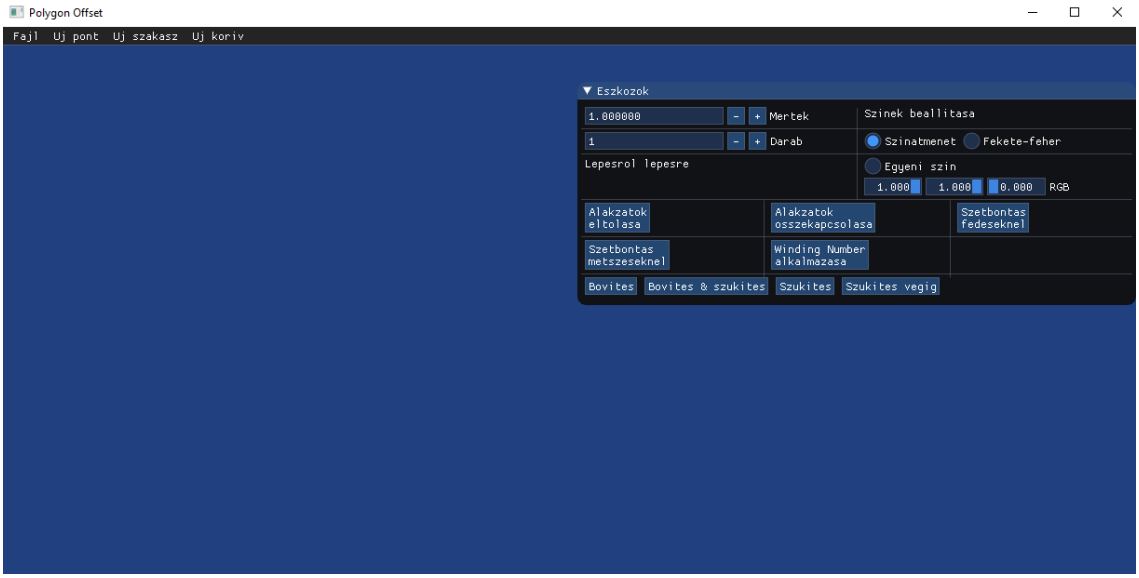
2.1 Telepítés, futtatás

A program nem igényel telepítést, a futtatható állományt elindítva használható. Mivel a program OpenGL 1.3 szabvány felhasználásával készült, ezért a futtató eszköznek rendelkeznie kell olyan grafikus illesztőprogrammal, ami támogatja azt.

Az alkalmazás csak Windows x86 és x64 változatokra lett lefordítva, de a C++ nyelv és az OpenGL API mind elérhetőek Unix-típusú rendszereken is, ezért a mellékelt forrásfájlok segítségével lefordíthatók bármely támogatott operációs rendszerre.

2.2 A fő program

A program indulásakor az 1. ábrán látható kezelőfelület jelenik meg. Ez a felület három részre bontható, a felső menüsávra, az Eszközök ablakra és az alkalmazás többi részét képező rajzfelületre.



1. ábra

2.3 A menüsáv

A menüsávban van lehetőségünk fájlból adatokat betölteni, illetve a programból adatokat kimenteni. Ezt a „Fajl” menü alatt megtalálható „Megnyitas” és „Mentes” gombokkal tehetjük meg. A fájl kiválasztását segíti egy külön megnyíló ablak, ahol ki tudunk választani egy tetszőleges dokumentumot a számítógépről.

- Új pont menü

Ha a menüben ezt az elemet választjuk, akkor a rajzfelületen a kurzor alatt megjelenik egy narancssárga kör, ezután a rajzfelület tetszőleges pontjára kattintva tudunk új pontot hozzáadni ahhoz. Ha előtte egy másik alakzat szerkesztése volt kiválasztva a menüből az előző alakzat szerkesztése félbeszakad. Jobb kattintás a rajzfelületen megszakítja az új pontok hozzáadását. Az utolsó két viselkedés hasonló a többi menüpontra is.

- Új szakasz menü

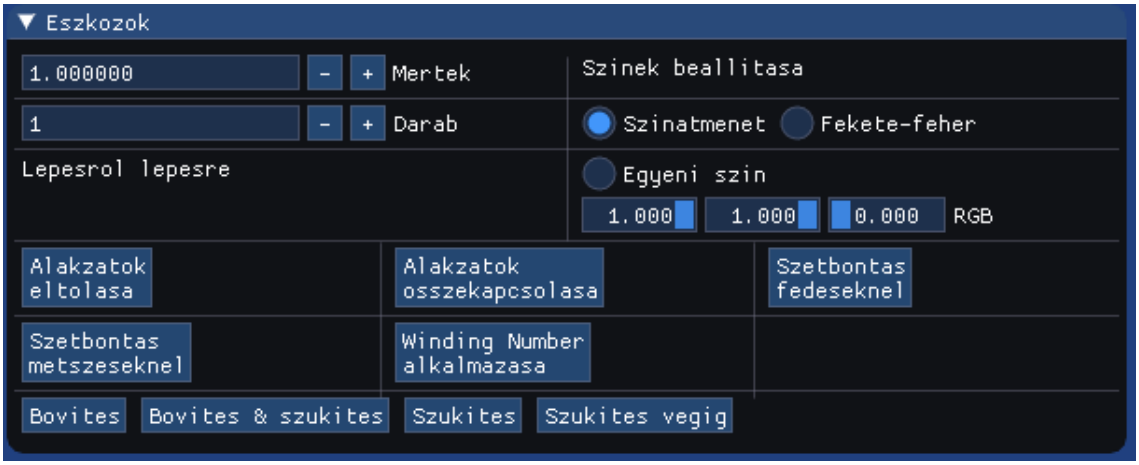
Ennek a menüpontnak a segítségével lehet új szakaszt felvenni a rajzfelületen. Miután kiválasztottuk ezt a menüpontot egy narancssárga kör jelenik meg a kurzor alatt a rajzfelületen, ekkor a szakasz kezdőpontját kell kijelölnünk. Ezt megadhatjuk egy új ponttal, ekkor elegendő bárhova kattintanunk a bal egérgombbal, vagy hozzákapcsolhatjuk egy másik ponthoz is a kezdőpontot.

A kapcsolás akkor fog létrejönni, ha elég közel visszük a kurzort az adott ponthoz, és annak színe pirosról narancssárgára vált. A szakasz végpontját a kezdőpontjához hasonló módon tudjuk megadni. Miután ez megtörtént a program a következő szakasz kezdőpontját várja, amíg meg nem szakítjuk a hozzáadást.

- Új körív menü

Ez a menüpont körívek hozzáadására szolgál. Először a körív középpontját kell kijelölni, majd kezdőpontját. A végpont meghatározása a középpontból a kurzor pozíciójába mutató vektor X tengellyel bezárt szögéből történik, illetve a középpont-kezdőpont távolságból. A körív sikeres hozzáadása után a program egy új körív hozzáadásába kezd.

2.4 Eszközök ablak



2. ábra

Az eszközök ablakban lehet az szűkítő-bővítő algoritmust paraméterezni, működését lépésről lépésre követni, illetve a rajzfelület színeit beállítani.

- Mérték

Egy valós szám bevitelére szolgáló mező. Ez a paraméter fogja meghatározni a szűkített/bővített poligon távolságát az eredeti poligontól.

- Darab

Egy egész szám bevitelére szolgáló mező. Értéke nagyobb, mint 0. Ettől a paramétertől fog függeni a szűkítés/bővítés iterációinak a száma, vagyis, hogy a kimenetbe maximum hány poligon kerülhet. A Szűkítés végig opció esetén nincs jelentősége.

- Alakzatok eltolása

Az algoritmus első lépését végzi el a rajzlapon található bemenet poligonra, ami ebben az esetben a poligon összes élének eltolását jelenti. Az eltolás irányát az Érték mezőben megadott szám előjel jelzi. Az eredményt grafikusan megjeleníti.

- Alakzatok összekapcsolása

Elvégzi az előző pontban leírt gomb funkcióját, majd az így kapott alakzatokat összekapcsolja egy később részletezett szabály szerint. Az eredményt grafikusan megjeleníti.

- Szétbontás fedéseknél

Elvégzi az előző két gomb funkcióját, majd az így kapott zárt alakzatban egymást fedő éleket keres, és ezek alapján létrehoz több olyan alakzatot, egy később leírt szabály alapján, amikben nincsenek egymást fedő élek. Az eredményt grafikusan megjeleníti.

- Szétbontás metszéseknél

Elvégzi az előző három gomb funkcióját, majd az így kapott zárt alakzatokban egymást metsző éleket keres, és ezek alapján létrehoz több olyan alakzatot, egy később definiált algoritmus alapján, amiben nincsenek egymást fedő alakzatok. Az eredményt megjeleníti grafikus formában.

- „Winding Number” alkalmazása

Elvégzi az előző négy gomb funkcióját, majd az így kapott zárt alakzatok megtartásáról vagy elvetéséről a később részletezett „Winding Number” algoritmus elvégzésével dönt. Az eredményt megjeleníti grafikusan.

- Bővítés

Elvégzi a teljes bővítő algoritmust a rajzlapon található poligonra. A bővítés mértékét a mérték mező abszolút értéke fogja meghatározni, az algoritmus iterációinak a számát pedig a darab mező határozza meg. Az eredményt megjeleníti grafikusan.

- Bővítés & szűkítés

Elvégzi a teljes bővítő, illetve szűkítő algoritmust a bemeneti poligonon. Az egyes algoritmusok mértéke a mérték mező abszolút értéke, darabszámuk pedig a darab mező értéke. Az eredményt megjeleníti grafikusan.

- Szűkítés

Elvégzi a szűkítő algoritmust a bemeneti poligonra. Az algoritmus mértéke a mérték mező abszolút értéke, maximális darabszáma pedig a darab mező értéke. Az eredményt megjeleníti grafikusan.

- Szűkítés végig

Elvégzi a szűkítő algoritmust a rajzlapon található poligonra. Az algoritmus mértéke a mérték mező abszolút értéke, viszont az előbbiekkel ellentétben figyelmen kívül hagyja a darab mező értékét, és addig végzi az algoritmust, amíg lehet tovább szűkíteni a poligont.

- Színek beállítása
  - Színátmenet

Az algoritmus eredményében lévő poligonok a rajztáblán a colors.txt fájlban definiált kulcsszínekből képzett színátmenettel lesznek színezve. Ha ezt az opciót választjuk, csak a választás után kirajzolt poligonokra lesz hatással.

  - Fekete-fehér

Ennél az opciónál a rajztábla háttérszíne fehérre, míg a rajztáblán kirajzolt bármely alakzat színe feketére vált. Az opció választása csak a választás után kirajzolt poligonokra lesz hatással.

  - Egyéni szín

Ennél az opciónál egyéni színezést lehet megadni az algoritmus kimenetelére az opciógomb alatt található csúszkák segítségével. Az opció csak a választása után felkerült poligonokra lesz hatással.

2.5 A rajzlap

- Mozgatás

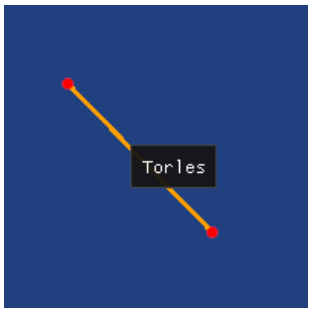
A rajzlap mozgatására a rajzlap egy szabad területén az egér jobb gombjának nyomva tartásával és az egér mozgatásával van lehetőség.

- Kicsinyítés és nagyítás

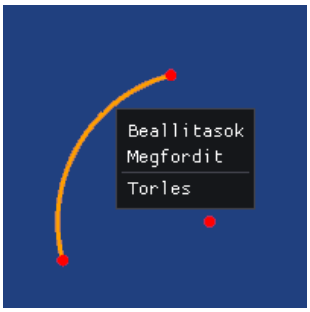
A rajzlap kicsinyítése és nagyítása az egér görgetésével lehetséges.

- Pont vagy alakzat lenyíló menü

Pont vagy alakzat lenyíló menüjének megnyitására akkor van lehetőség, ha a kurzor a pont vagy alakzat felett áll, és annak narancssárga színe van. Ekkor egy jobb kattintással hozható elő a lenyíló menü.



3. ábra



4. ábra

5. ábra

  - Szakasz lenyíló menü (3. ábra)

A szakasz lenyíló menüjében csak a szakasz törlésére van lehetőség. Erre a gombra kattintva törlődik a szakasz, illetve azon végpontjai, amelyekhez nem kapcsolódik más alakzat.

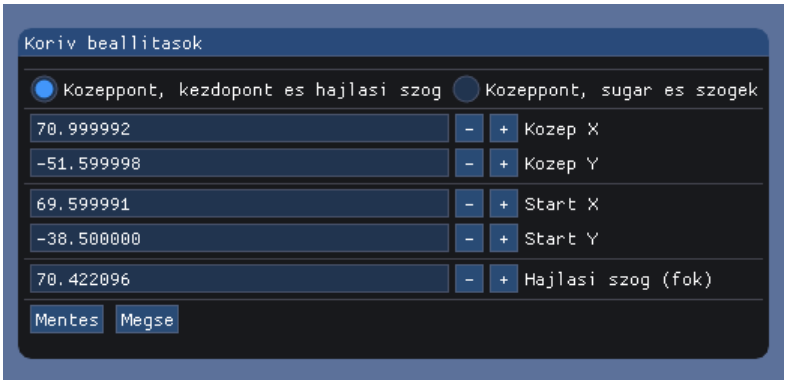
  - Körív lenyíló menü (4. ábra)

A körív lenyíló menüjében lehetőség van a körív egyes paramétereinek beállítására a beállítás menüpontra kattintva, ami egy külön felugró ablakot hoz elő, továbbá a körív megfordítására illetve törlésére. A törlés a szakasz törléséhez hasonlóan történik.

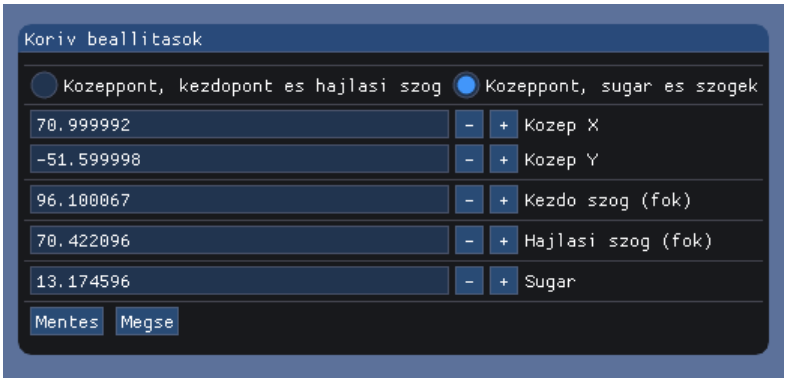
- Pont lenyíló menü (5. ábra)

A pont lenyíló menüjében lehetőségünk van a pont pozíciójának módosítására a beállítások menüpont alatt, ami egy külön felugró ablakot hoz elő. Továbbá törölhetjük is a pontot a törlés menüponttal. A pont törlése az összes hozzá kapcsolódó alakzat törlését vonja maga után.

- Felugró ablakok
  - Körív felugró ablak



6. ábra



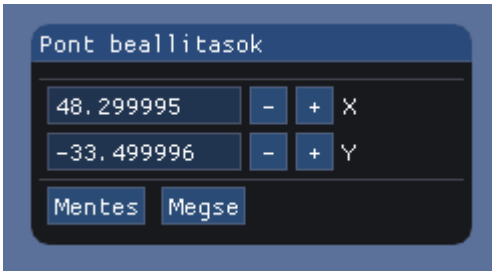
7. ábra

A körív felugró ablakban módosíthatjuk a körív középpontjának x és y koordinátáit, kezdőpontjának koordinátáit, valamint a hajlási szögét fokban megadva, ami lehet negatív is (6. ábra).

Ezen felül a körívet leírhatjuk a középpontja koordinátáival, sugarának hosszával, valamint a középpontjából a kezdőpontjába mutató vektor x tengellyel bezárt szögeként értelmezett kezdőszöggel és a kezdő szögtől számított hajlásszöggel. Csak a hajlászög lehet negatív (7.ábra).

A két adatbeviteli mód között az ablak felső sávjában tudunk váltani.

- Pont felugró ablak (8. ábra)



8. ábra

A pont felugró ablakban módosíthatjuk az adott pont x és y koordinátáit. A mentés gombra kattintva tudjuk elmenteni a változtatásokat. Ha így tettünk, a ponthoz kapcsolt összes alakzat frissülni fog.

3. Fejlesztői dokumentáció

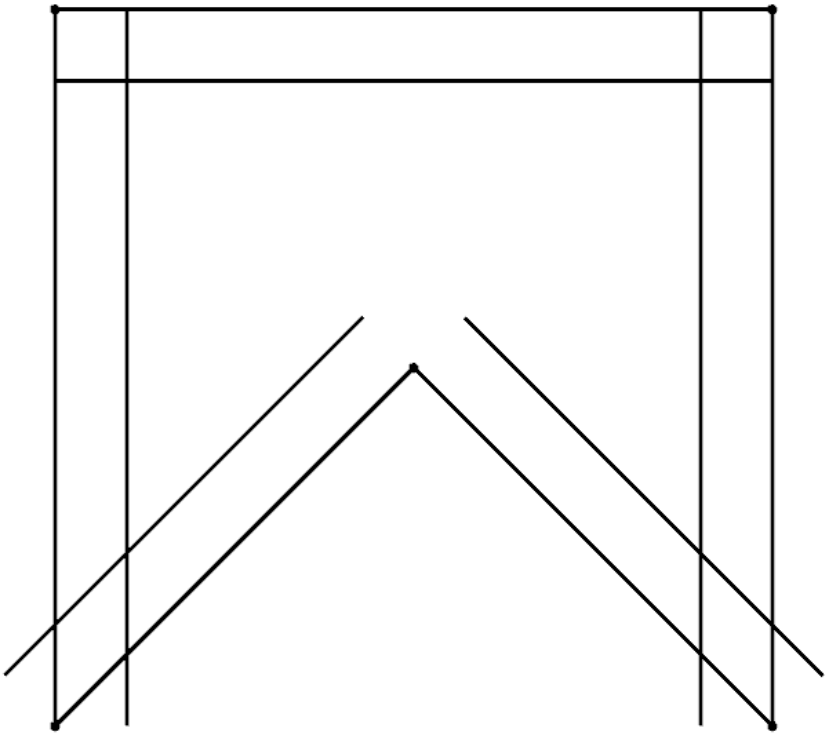
A következőkben részletezem a feladat megoldó algoritmusát, betekintést adok az algoritmust kulcsfontosságú lépéseibe, valamint bemutatom a program felépítését.

3.1 Az algoritmus

Az algoritmus hat kulcsfontosságú lépésből áll, amit a 9. ábra szemléltet. Ezek a lépések egymásra épülnek, mindegyik lépés az őt megelőző lépés eredményével dolgozik tovább. A bemenet egy darab poligon, aminek nincsen egymást fedő vagy keresztező éle, továbbá a körbejárási iránya óramutató járásával ellentétes. A kimenet lehet nulla, egy vagy több poligon, melyekre ugyanazok a tulajdonságok igazak, mint a bemeneti poligonra.

9. ábra

3.1.1 Alakzatok eltolása



10. ábra

Ennek a lépésnek a célja a poligon alakzatainak eltolása egy adott mértékkel, a 10. ábrán látottakhoz hasonló módon. Az eltolás olyan alakzatokat képez, amiknek minden pontja adott távolságra fekszik az eredeti alakzat pontjaitól.

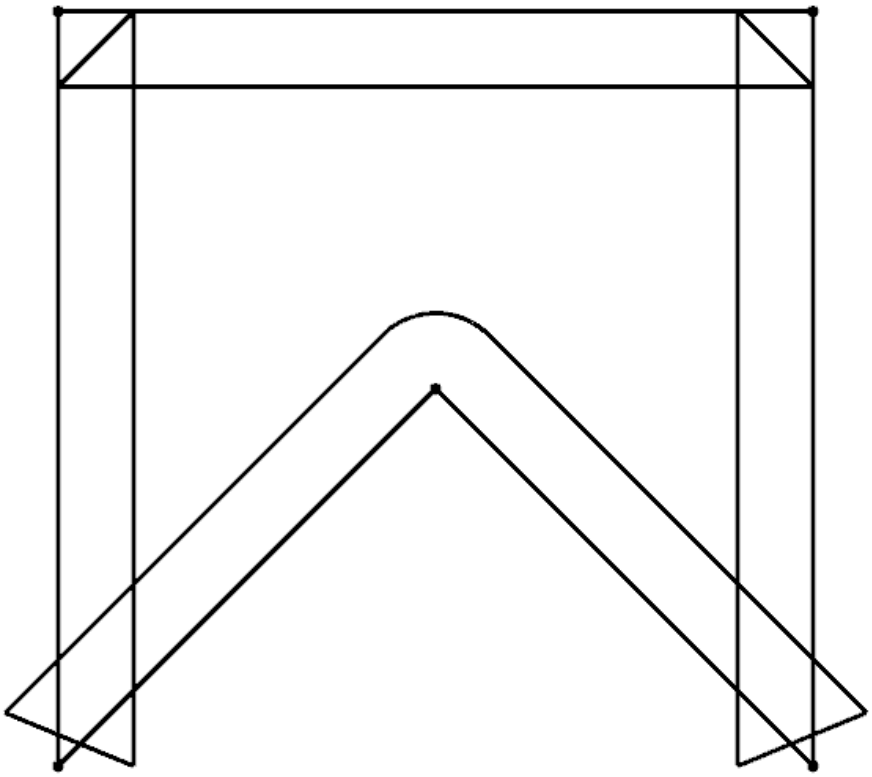
- Szakasz eltolása

Az eltolás művelet szakasz esetében a szakasz normálvektorának irányába történik, ami a szakasz irányvektorából számítandó annak x és y koordinátái felcserélésével, majd az így kapott vektor x koordinátájának negálásával. Ezek után egyszerűen beszorozzuk az egység hosszú normálvektort az eltolás mértékével, majd hozzáadjuk a szakasz kezdő- és végpontjához. Az így kapott pontok összekötésével kapjuk az eltolt szakaszt.

- Körív eltolása

Körív esetén csökkentjük, vagy növeljük annak sugarát az eltolás mértékével. Az így kapot sugárral és az eredeti kezdő és hajlásszöggel kiszámítjuk az eltolt körív új kezdő és végpontját, majd ezeket összekötve, az eredeti körív irányában, kapjuk az eltolt körívet. Ha az eltolásnál a körív sugara negatív szám lett az eltolt pontokat az eredeti irány ellentetjében kötjük össze, mert ilyenkor a körív megfordul. Ha a sugár egy nullához közeli szám megtartjuk az eredeti paramétereit a körívnek, kivéve a sugarat, mert a következő lépésben még hasznosak lesznek a szögek.

3.1.2 Alakzatok összekapcsolása



11. ábra

Ez a lépés az előző művelet eredményével dolgozik, és plusz alakzatokat szúrhat be a poligon alakzatai közé, ha szükséges, egy zárt kör létrehozásának érdekében (11. ábra).

A beszúrás az eltolt alakzatok viszonyától függ. Csak a szomszédos alakzatok viszonyát kell vizsgálni, mivel azok alkottak az eltolás előtt egy zárt kört. Ez a viszony háromféleképp alakulhat.

- A szomszédos alakzatok kezdő- és végpontja egybeesik

Ebben az esetben nincs semmi teendő, mivel a kör folytonossága nem szakad meg.

- A szomszédos alakzatok kezdő- és végpontja nem esik egybe, valamint metszéspontjuk sincs

Ebben az esetben a két alakzat eltávolodott egymástól. Mivel a szűkítés vagy bővítés célja, hogy az alakzatok a művelet után adott távolságra legyenek, ezért ebben az esetben a két alakzat közé egy körív beszúrásával kell kiegészítenünk a poligont, ugyanis a körív minden pontja ugyanolyan távolságra van a középpontjától, ami ebben az esetben a két alakzat kezdő- és végpontja az eltolás előttről.

A körív pontos adatainak számítása a kezdő és végpontjából, illetve az őt megelőző alakzat végpontjában, valamint az őt követő alakzat kezdőpontjában található normálvektorokból történik. A pontokból és az azokhoz tartozó normálvektorokból két egyenest képzünk, úgy, hogy az egyenesek irányvektora lesz a normálvektor. Az így kapott egyeneseket elmetszük egymással. A metszéspontjuk adja a körív középpontját. Ha a két egyenes párhuzamos, a középpont a kezdő- és végpontok között félúton lévő pont lesz. A középpont ismeretében a körív többi tulajdonsága már könnyen számítható.

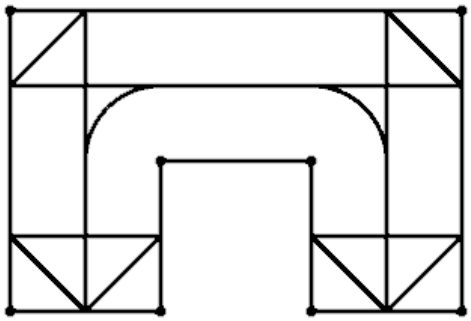
- A szomszédos alakzatok kezdő- és végpontja nem esik egybe, de van metszéspontjuk

Ebben az esetben a két alakzat egybecsúszott. Összekötésük módja lényegtelen, ezért a legegyszerűbb módon egy szakasz beszúrásával egészítjük ki a poligont a szakadás megszüntetéséhez. Így létrejön egy hurok a poligonban, de ez a későbbi lépések során el fog tűnni a végeredményből.

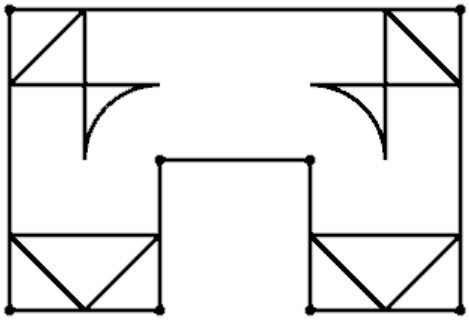
3.1.3 Összecsukló alakzatok eltávolítása

Az alakzatok eltolása és összekapcsolása után előfordulhat, hogy a poligon tartalmaz nulla vagy ahhoz nagyon közeli hosszúságú alakzatot. Ezeket el kell távolítani a poligonból, mivel már semmilyen hasznuk nincs, az összecsukló alakzatok szomszédjainak kezdő- és végpontja egybeesik, és nélkülük a poligon zártsága is megmarad, valamint a későbbi számításoknál bezavarhatnak.

3.1.4 Szétbontás egymást fedő alakzatok alapján



12. ábra



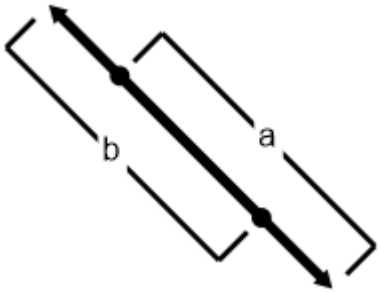
13. ábra

Ebben a lépésben a nem szomszédos alakzatok közti átfedéseket vizsgáljuk. Két alakzat csak akkor fedheti egymást, ha a típusuk megegyezik, tehát szakasz csak szakaszt fedhet, körív csak körívet.

Fontos, hogy szakaszok esetén csak akkor jöhet létre átfedés, ha a két szakasz párhuzamos, és normálvektoruk ellentétes irányú. Ha normálvektoraik megegyeznének akkor az eltolás során ugyan abba az irányba mozdulnak el, és mivel feltételezzük, hogy az eltolás előtt nem fedtéd egymást, utána sem fogják. Körívek esetében a középpontjuk egybe kell hogy essen, sugaruk meg kell hogy egyezzen, továbbá hajlási szögük előjelének ellentétesnek kell lennie. Ennek indoklása a szakaszok esetével hasonló.

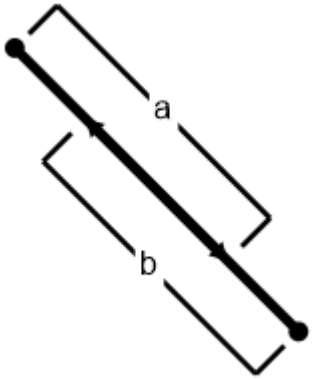
Feltehető, hogy kettőnél több alakzat sohasem fogja fedni egymást, ugyanis kettőnél több alakzat esetén biztosan lesz legalább két olyan, amiknek a normálvektorai azonos irányba mutatnak, és ekkor ugyanabba az irányba tolódnak el, de mivel feltesszük, hogy az eltolás előtt nem fedték egymást, ezért továbbra sem fogják.

- Szakasz átfedés



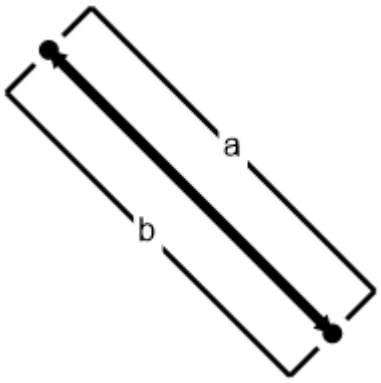
14. ábra

Az „a” szakaszt a „b” szakasz a kezdőpontjában fedi (14. ábra). Ebben a helyzetben az „a” szakaszt a „b” szakasz kezdőpontjától az „a” szakasz végpontjáig kell tekinteni, illetve a „b” szakaszt az „a” szakasz kezdőpontjától a „b” szakasz végpontjáig, tehát a poligonban a „b” szakasz kezdőpontjába befutó alakzat ezzel az új alakzattal folytatódik, illetve az „a” szakasz kezdőpontjából a „b” szakasz végpontjába haladunk tovább.



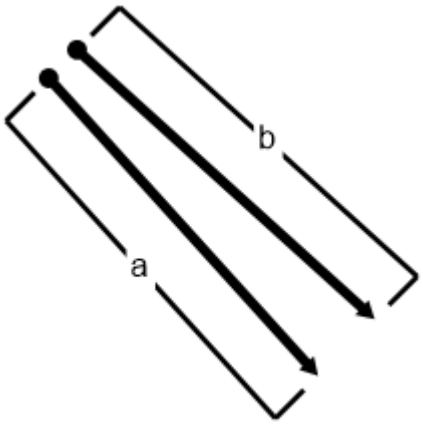
15. ábra

Az „a” szakaszt a „b” szakasz a végpontjában fedi (15. ábra). Ebben az esetben az „a” szakasz annak kezdőpontjától a „b” szakasz végpontjáig tekintendő, illetve a „b” szakasz annak kezdőpontjától az „a” szakasz végpontjáig, tehát a poligonban az „a” szakasz a „b” szakaszt követő alakzattal folytatódik, a „b” szakasz pedig az „a” szakaszt követővel.



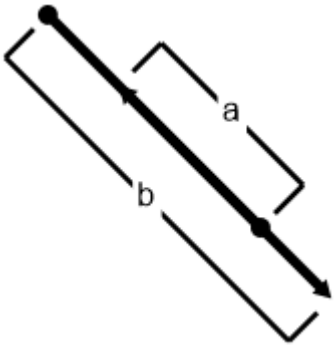
16. ábra

Az „a” és „b” szakaszok teljes hosszukban lefedik egymást (16. ábra). Ebben az esetben a poligonban az „a” szakaszt a „b” szakaszt követő alakzat követi, valamint a „b” szakaszt az „a” szakasz utáni alakzat követi.



17. ábra

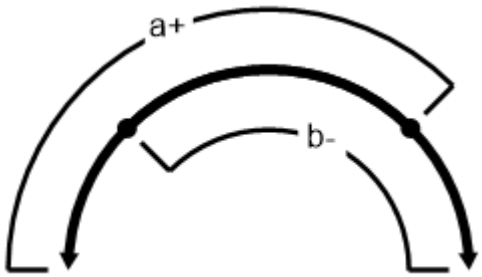
Az „a” és „b” szakaszok nem fedik egymást (17. ábra). Ebben az esetben nincs semmi teendő, a poligonban az alakzatok sorrendje marad az eredeti.



18. ábra

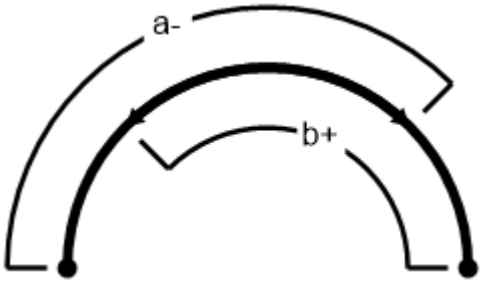
A „b” szakasz teljesen lefedi az „a” alakzatot (18. ábra). Ebben az esetben az „a” szakasz jelentéktelennek tekinthető, és az őt megelőző alakzatot rögtön az őt követő fogja követni. A „b” szakasz két részre esik, az egyik a „b” startpontjából „a” végpontjába visz, ahonnan az „a” szakaszt követő alakzat érvényes, a másik az „a” kezdőpontjából a „b” végpontjába haladó szakasz lesz. Itt a következő alakzat marad a „b” utáni.

- Körív átfedés



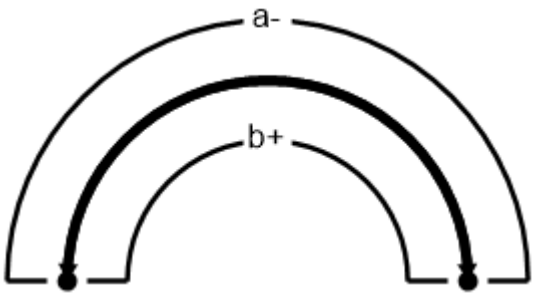
19. ábra

A „b” körív az „a” körívet a kezdőpontjában fedi (19. ábra). Ez az eset a 14. ábránál részletezett szakasz kezdőpont-átfedéshez hasonló.

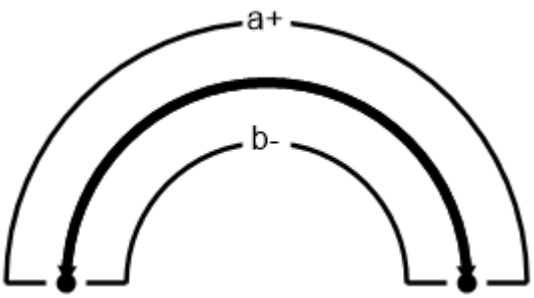


20. ábra

A „b” körív az „a” körívet a végpontjában fedi le (20. ábra). Ez az eset a 15. ábránál részleteszett szakasz végpont-átfedéshez hasonló.

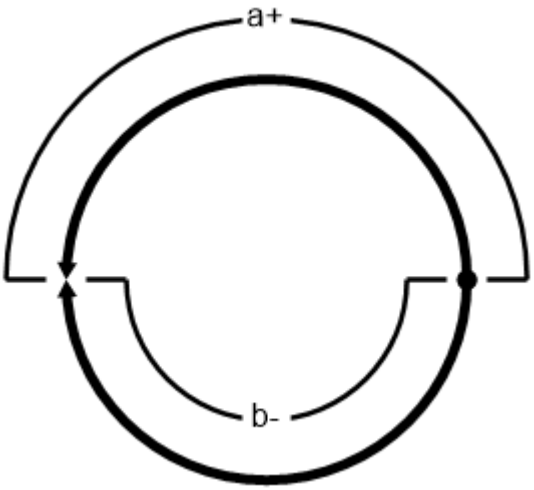


21. ábra

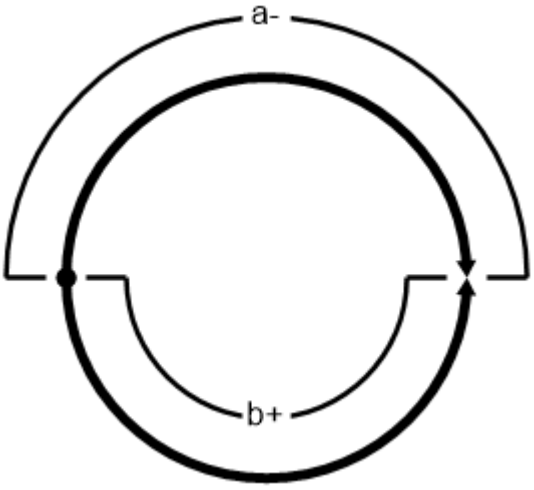


22. ábra

Az „a” és „b” köívek teljes egészükben lefedik egymást (21. és 22. ábrák). Ez az eset a 16. szakasz teljes-átfedéshez hasonló.

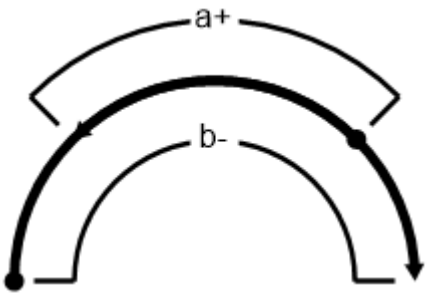


23. ábra

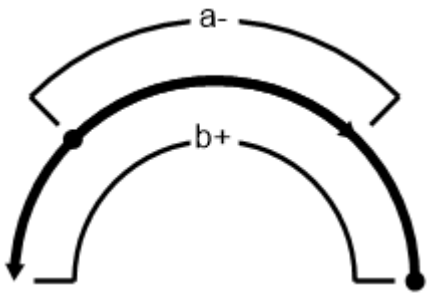


24. ábra

Az „a” és „b” körívek nem fedik egymást. A 23. és 24. ábrákon lévő két speciális esetet a körívek kezdő- és hajlászszögének vizsgálatával lehet detektálni. Ha a kezdőszögek megegyeznek és a kezdőszögek és hajlászögek összegeként kapott szögek értéke azonos, akkor a két körív biztosan nem fedi egymást. Ez az eset általánosságban hasonló a 17. ábránál részletezett szakasz nem-fedéssel.

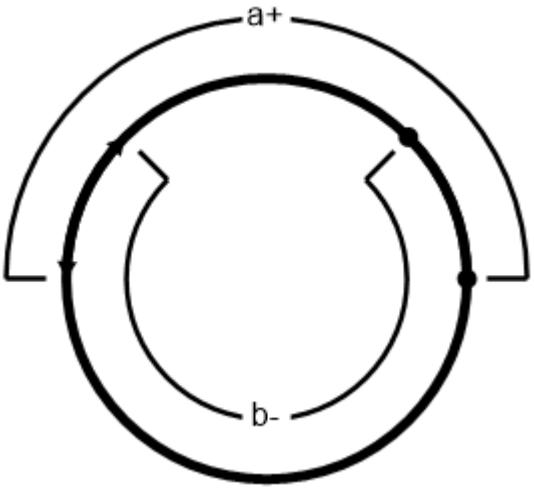


25. ábra

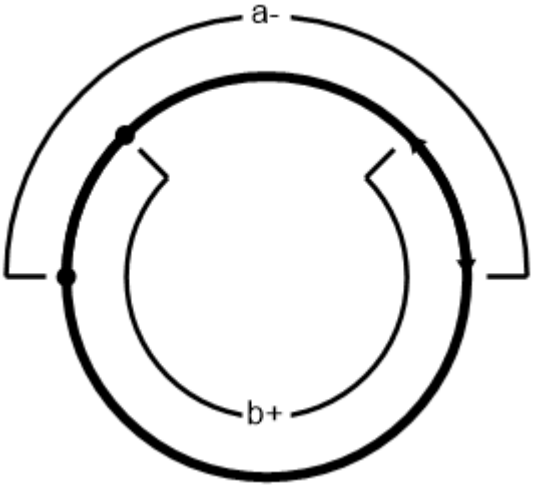


26. ábra

A „b” körív lefedi az „a” körívet teljes egészében (25. és 26. ábrák). Ez az eset hasonló a 18. ábránál részletezett szakasz lefedéssel.



27. ábra

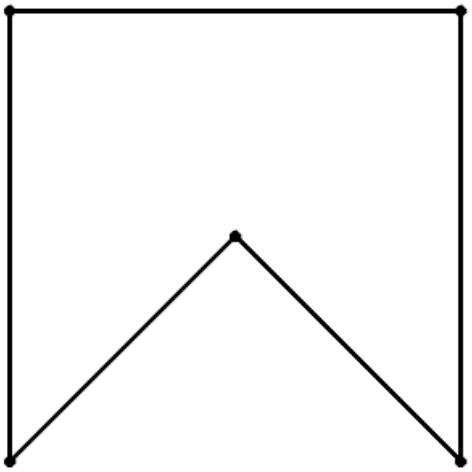


28. ábra

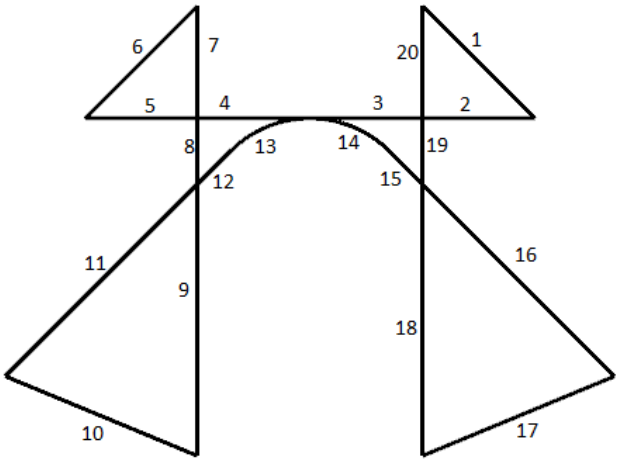
Az „a” körív a „b” körívet a keződ- és végpontjában is lefedi, de nem fedi le a teljes „b” körívet (27. és 28. ábrák). Ebbel az esetben az „a” körív a „b” körív kezdőpontjától a „b” körív végpontjáig tekintendő. Hasonló módon a „b” körív az „a” körív kezdőpontjától az „a” körív végpontjáig nincs átfedésben. A poligon „a” köríc kezdőpontjába tartó élét a „b” körív nem-lefedett része követi, majd az „a” körívet követő alakzat. Az új követési sorrend a „b” körív esetén is hasonló.

3.1.5 Szétbontás egymást keresztező alakzatok alapján

A szűkítés során egymást keresztező élek, hurkok keletkezhetnek a poligonban (30. ábra). Ezek alapján a poligon több részpoligonra bontható, ahol már nincsenek ilyen típusú alakzatok. Ez a lépés az ilyen részpoligonok kiszűrésére szolgál. A kiszűrt poligonokra továbbá az is igaz, hogy a körbejárási irányuk az óramutató járásával ellentétes.

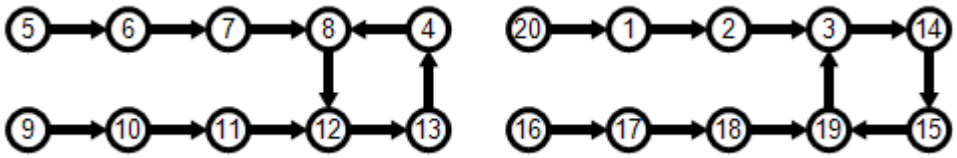


29. ábra



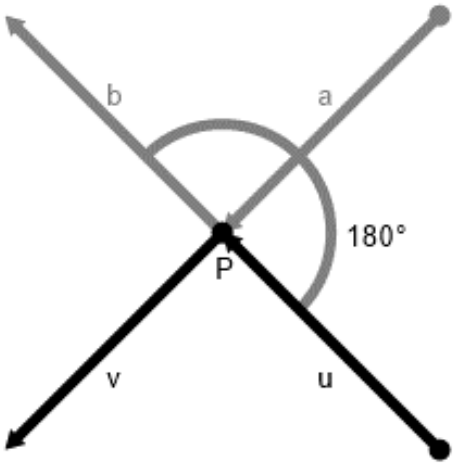
30. ábra

Először meg kell vizsgálni a poligon éleit metszések szempontjából. Szakaszok esetén csak a nem szomszédosok vizsgálata szükséges, de ha a két szomszédos alakzat közül az egyik nem szakasz, akkor őket is vizsgálni kell, hogy van-e közös pontjuk a kezdő- és végpontjukon kívül. Fontos, hogy a fedéssel ellentétben egy pontban akár több alakzat is metszheti egymást.

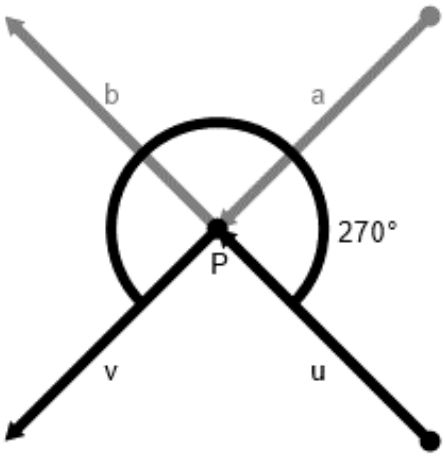


31. ábra

Az így keletkezett pont- és élhalmazra tekinthetünk úgy, mint egy irányított gráfra, ami geometriai jelentéssel is bír (30. ábra). Ebből a gráfból képezzünk egy másik gráfot, aminek a csúcsai ennek a gráfnak az élei. Legyen ez az élkövetési gráf, ami szintén irányított gráf. Legyen U, V csúcsok ebben a gráfban, azaz U és V a poligon élei közül valók (31. ábra).

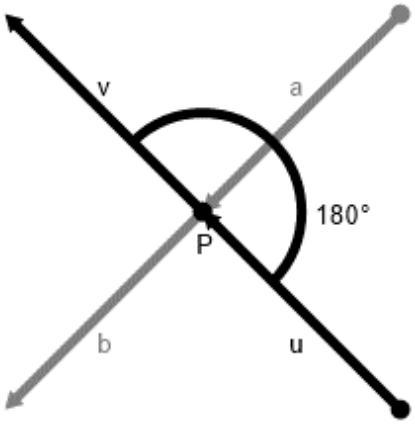


32. ábra

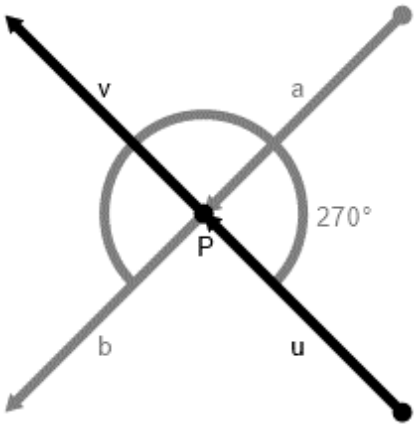


33. ábra

Szűkítéskor az élkövetési gráfban van él u-ból v-be, ha a poligon gráfjában létezik olyan P csúcs, hogy az u-nak végpontja v-nek pedig kezdőpontja, továbbá u és v bezárt szöge maximális, azaz u a P csúcs élei közül a legnagyobb szöget zárja be óramutató járásával ellentétes irányba az olyan élek közül, amiknek a kezdőpontja P-ben van (33. ábra).

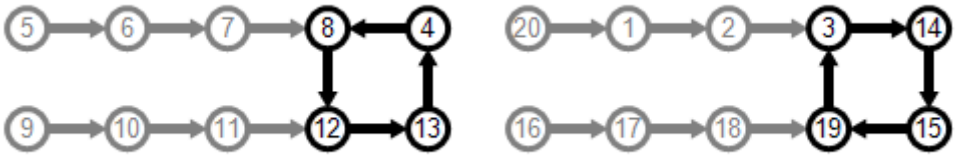


34. ábra

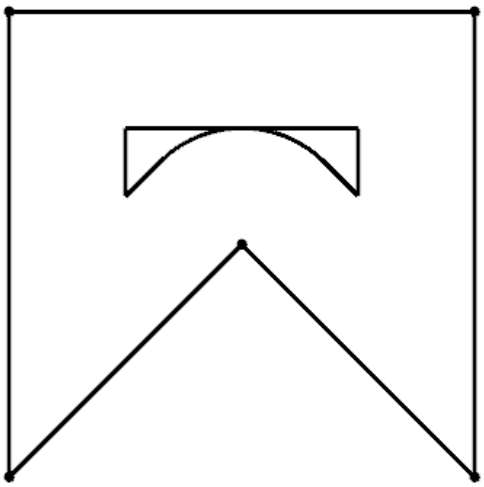


35. ábra

Bővítéskor a szűkítéshez hasonló módon építjük fel az élkövetési gráfot, azzal a különbséggel, hogy a minimális szögeket keressük a poligon gráfjában annak élei közt (34. ábra).



36. ábra

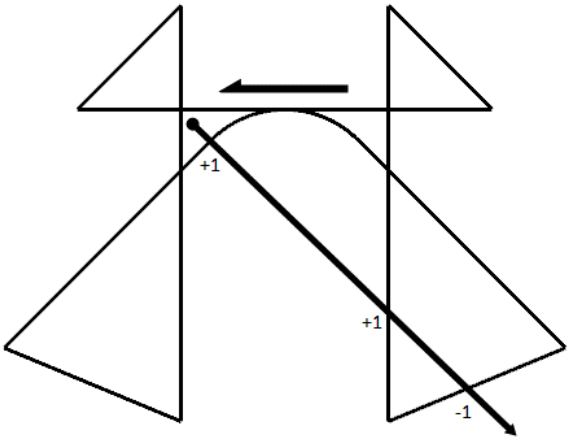


37. ábra

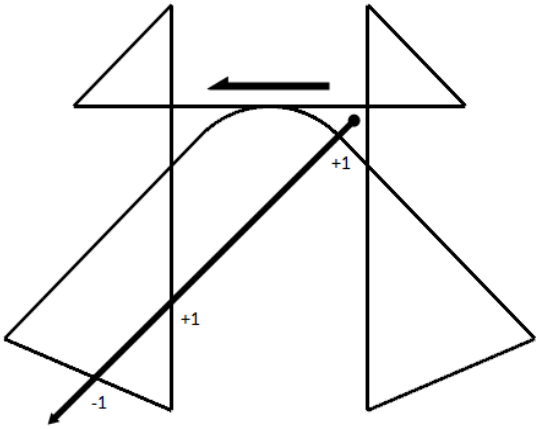
Az így kapott élkövetési gráfban ha megkeressük az összes kört (36. ábra), és a kört alkotó élekből poligonokat építünk (37. ábra), akkor ezekre igaz lesz az, hogy a körbejárási irányuk óramutató járásával elentétes, valamint nincs bennük élkereszteződés.

3.1.6. „Winding Number” algoritmus alkalmazása

A „Winding Number” algoritmust az olyan poligonok kiszűrését szolgálja, amiknek a körbejárási iránya az óramutató járásával ellentétes, mégsem részei az eredménynek.



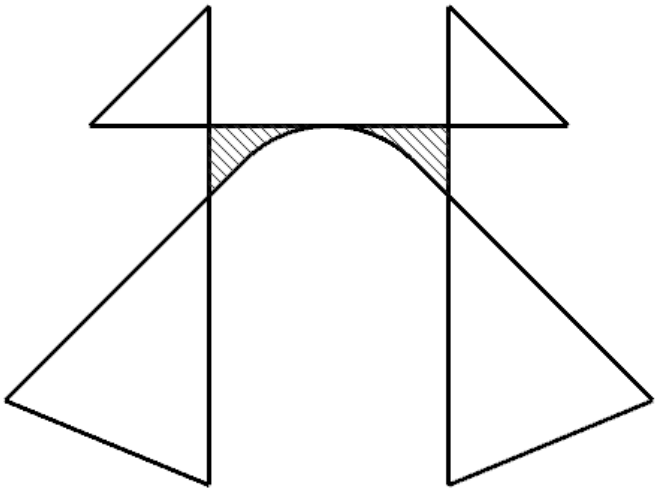
38. ábra



39. ábra

A „Winding Number” algoritmus bemenete egy élhalmaz, ami csak szakaszokat tartalmazhat, valamint egy félegyenes, ami nem metszi egyik szakaszt sem annak kezdő- vagy végpontjában. A félegyenessel minden szakaszra megvizsgáljuk, hogy van-e metszés. Ha van, és a félegyenes kezdőpontja balra esik a szakasztól, akkor növeljük a számlálót egyel. Ha a félegyenes kezdőpontja jobbra esik a szakasztól, csökkentjük egyel (38. és 39. ábrák). Az így kapott egész szám azt jelenti, hogy a poligon hányszor kerüli meg a félegyenes kezdőpontját.

A „Winding Number” algoritmust az eljárás harmadik lépéseként kapott polgonra kell alkalmazni, azaz az eltolt, összekapcsolt nulla hosszú alakzatok nélküli poligonra. Ha ez a poligon tartalmaz köríveket azokat valahány szakasszal helyettesíteni kell. A félegyenes kezdőpontjának meghatározására szolgálnak az előző lépésben kapott részpoligonok, amikben nincsenek egymást keresztező vagy fedő élek. Minden ilyen poligonra le kell futtatni a „Winding Number” algoritmust a bekezdés elején említett poligonra, valamint olyan félegyenesekre amiknek a kezdőpontja a részpoligon bármely pontja.



40. ábra

Az eredménybe szűkítéskor azok a poligonok kerülhetnek bele, amikre a „Winding Number” algoritmus eredménye 1, tehát az eredeti poligon területének a része, és nem azon kívüli (40. ábra).

Bővítéskor mindig lesz egy olyan poligon, a legkülső, amely benne lesz az eredményben, mivel végtelen a terünk ahová bővíthetünk. A belső poligonok közül azokat kell megtartani, amiknek a „Winding Number” értéke nulla, mivel bővítésnél mindig a legkisebb bezárt szögű alakzatokra léptünk, ezért olyan poligonokat kapunk amik nem részei az eredeti poligonnak.

3.2 A program felépítése

- Sketch

A programban a felhasználó által szerkesztett vagy betöltött poligon vázlat tárolására szolgál. A jelenlegi implementációban csak egy vázlatot lehet kezelni, de ez könnyen módosítható több példány kezelésére is ugyan azon programon belül. Definiálva a „Sketch.h”, implementálva a „Sketch.cpp” állományokban van.

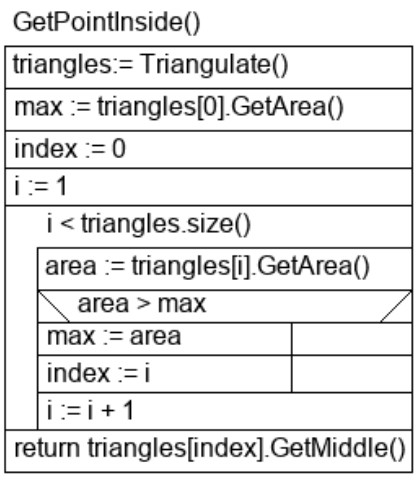
A betölthető fájl kiterjesztése bármilyen lehet, de a formátuma kötött. Első sora egy egész számot tartalmaz. Ez határozza meg a betöltendő csúcsok számát. Ezt követi a csúcsok adatainak sorai. Minden csúcs sorában annak x és y koordinátái vannak eltárolva valós számként, a törtrész tizedes ponttal, a két koordináta érték pedig pontoszvesszővel elválasztva. A csúcsok adatai után az alakzatok száma jön egy sorban, ami egy egész szám. Ez a sor után jönnek az alakzatok adatai. Minden sor elején az alakzat típusa szövegesen szóközzel elválasztva a sor második felében lévő adatoktól. A típus lehet „Segment” vagy „Arc”. Szakasz esetén a sor második felében lévő adatok tartalmazzák a szakasz kezdő- és végpontjának indexét pontosvesszővel elválasztva a feljebb definiált csúcslistából. Körív esetén a sor második felében az adatok a körív kedző-, vég- és középpontjának indexét tartalmazzák a csúcslistából, valamint egy igaz-hamos értéket. Ez igaz, ha a körív hajlásszöge negatív. Mindez pontosvesszővel elválasztva.

„ToPolygons” nevű funkciójával kérhető le az összes zárt alakzat listája poligonoként. Ebből a listából a legelső poligon lesz az algoritmus bemenete, feltéve hogy van elem a listában.

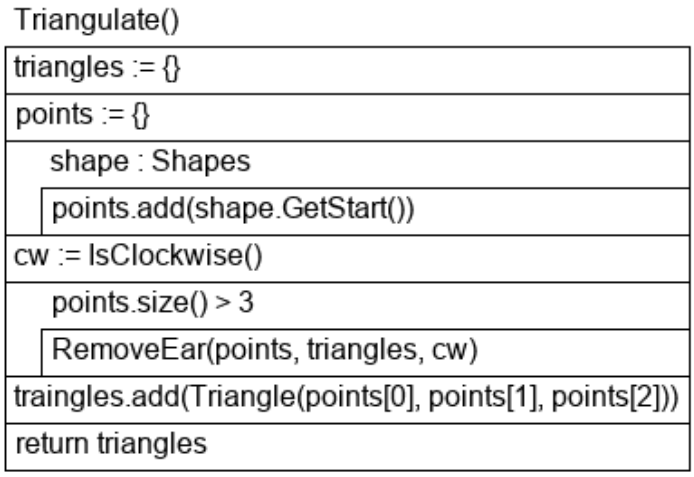
- Polygon2

Egy kétdimenziós poligon adatainak tárolására szolgáló osztály. Definíciója a „Polygon2.h”, implementációja a „Polygon2.cpp” állományokban található. Az algoritmus lépései ebben az osztályban vannak megvalósítva statikus metódusokként.

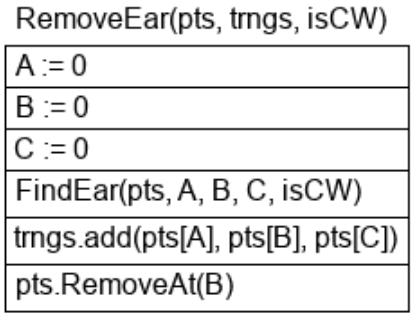
Kiemelendők az utolsó lépéshez félegyenest generáló „GetNoIntersectingRay”, valamint a poligon egy belső pontját kiszámító „GetPointInside()” (41. ábra) funkciók. A belső pont kiszámításához először helyettesítjük a poligonban található nem lineáris alakzatokat szakaszokkal, maly az így kapott poligont háromszögesítjül és kiválasztjuk a legnagyobb területtel rendelkező háromszöget. E háromszögnek a középpontja biztosan benne lesz a poligonban.



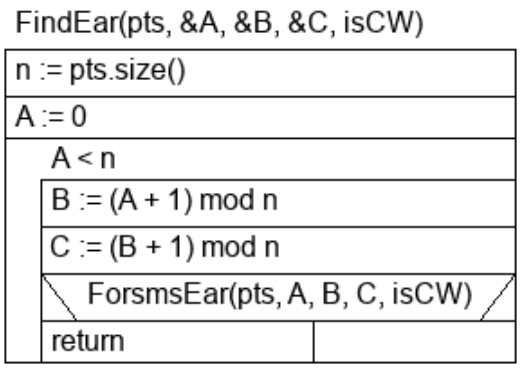
41. ábra



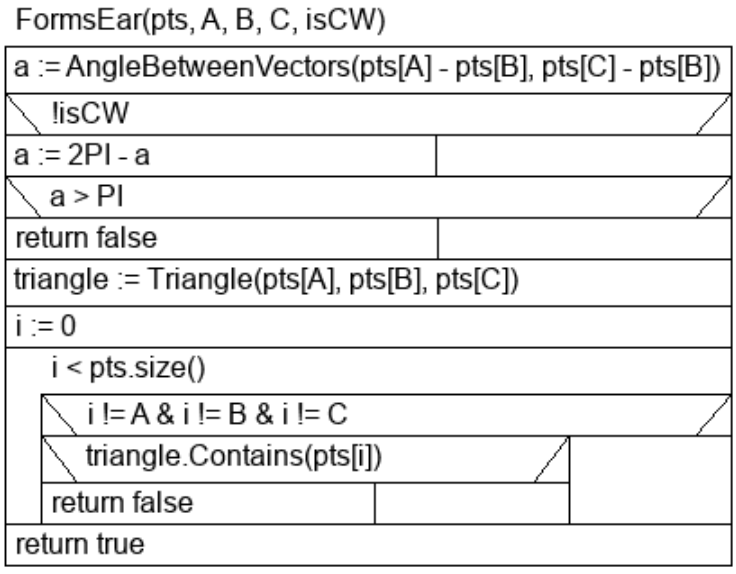
42. ábra



43. ábra

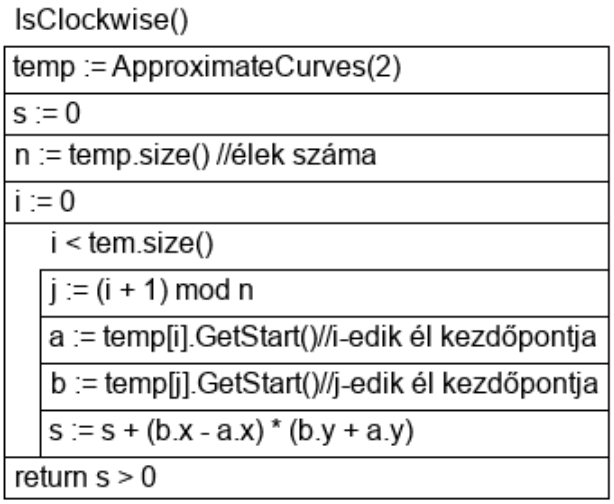


44. ábra

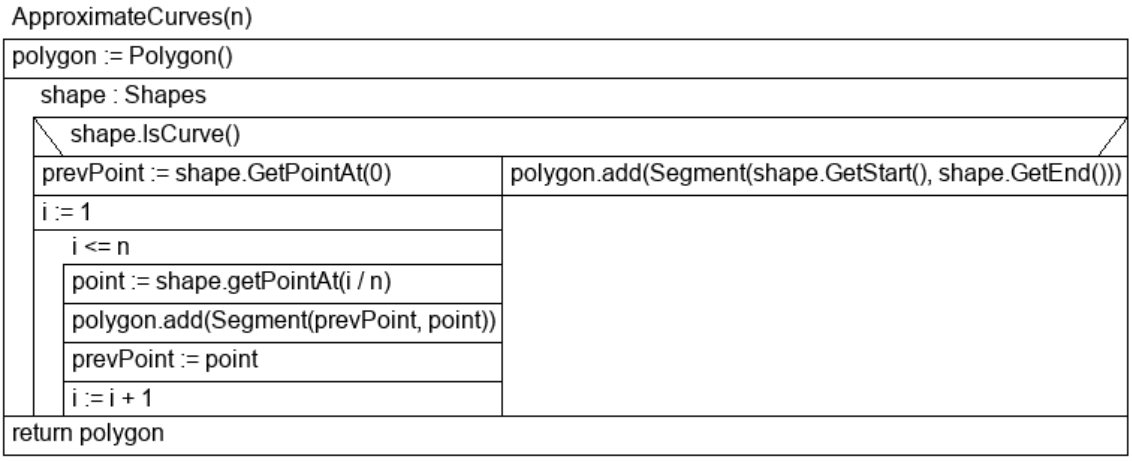


45. ábra

Ha megvan a félegyenes kezdőpontja, meg kell határoznunk az irányát. Ehhez a félegyenes kezdőpontjából a poligon egyes csúcsaiba mutató vektorok x tengellyel bezárt szögeire van szükségünk. Az egymástól legtávolabb eső szögek számtani közepe lesz a optimális irány, mivel ez fog legtávolabb haladni bármely csúcstól.

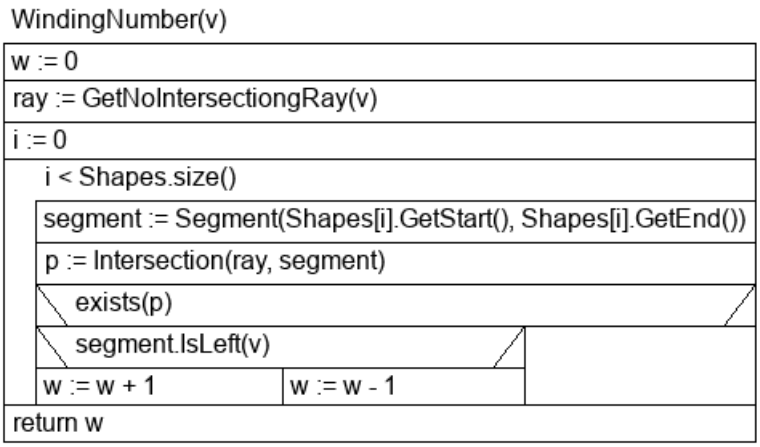


46. ábra



47. ábra

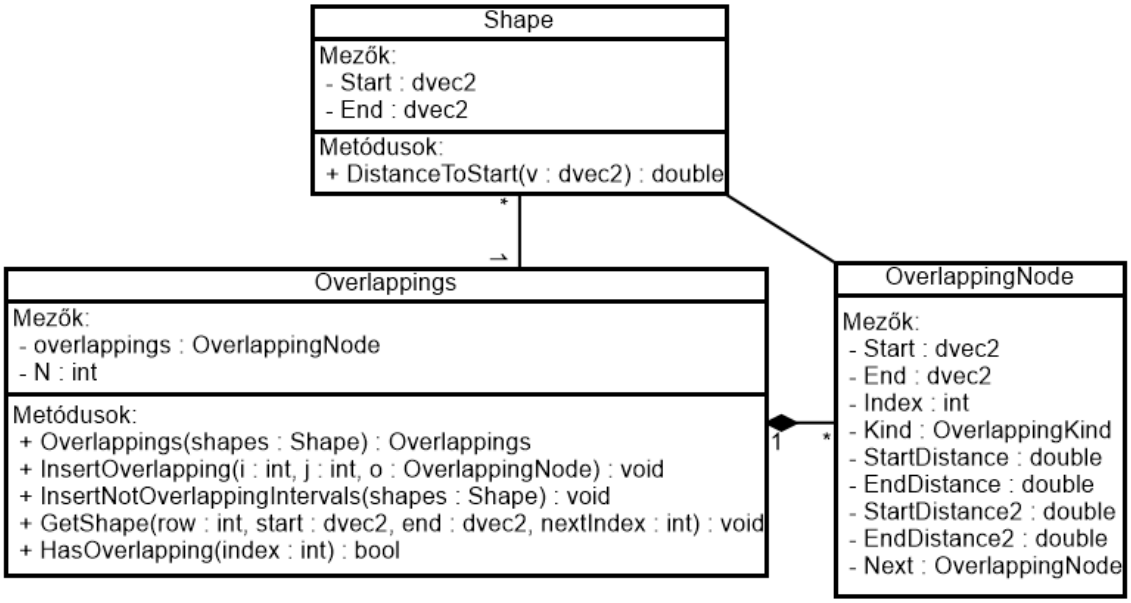
A „Winding Number” algoritmus implementációja.



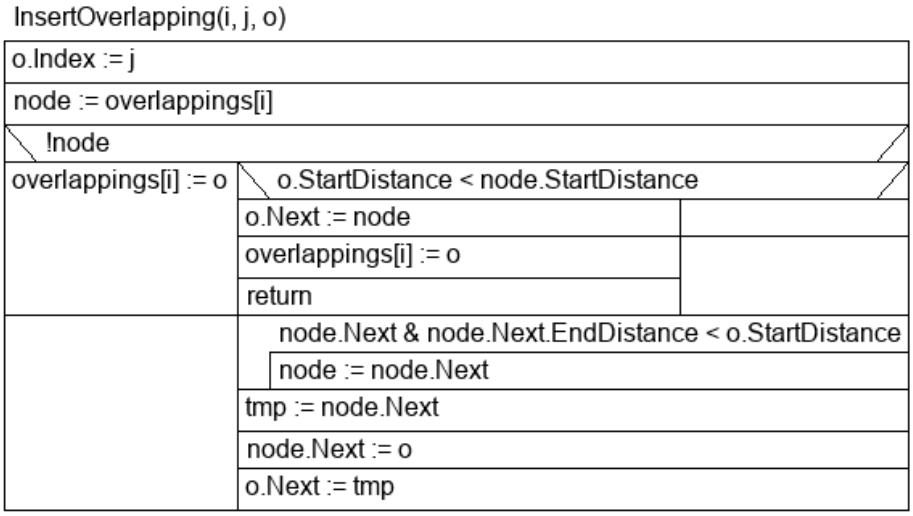
48. ábra

- OverlappingSplitter

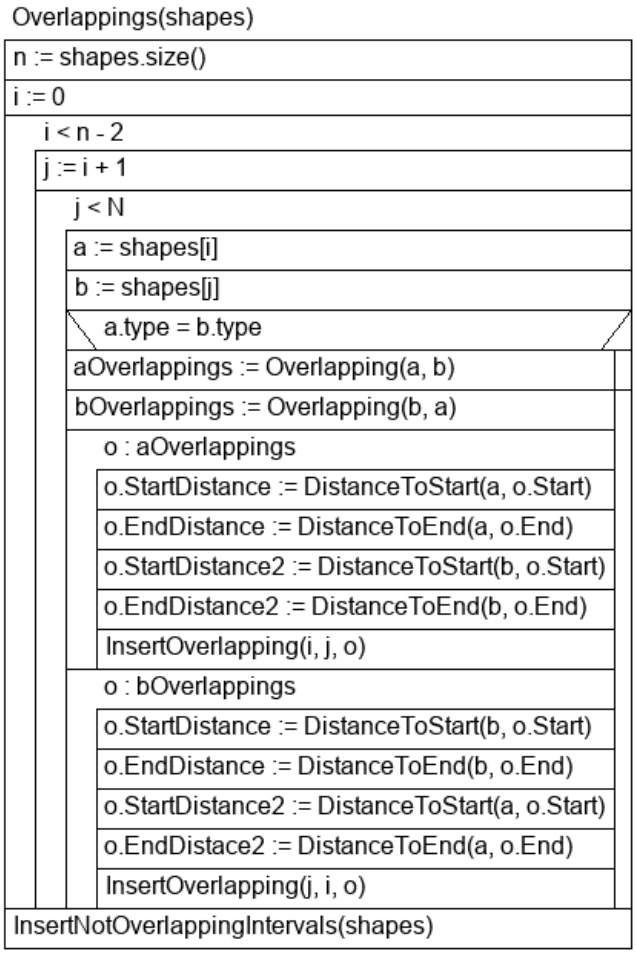
A poligon fedések szerinti szétbontására szolgáló segédosztály. Tartalmaz egy tömböt, aminek elemszáma megegyezik a poligon éleinek számával. A tömb minden eleme egy rendezett láncolt lista kezdő eleme. A láncolt listák „a” és „b” pontokból álló intervallumokat tartalmaznak, amik a fedéseket illetve a nem-fedéseket szimbolizálják az alakzat kezdetétől a végéig. Minden intervallum rendezve van növekvő sorrendbe az alakzat és az intervallum kezdőpontjának távolsága szerint. A távolság itt az alakzat mentén értendő.



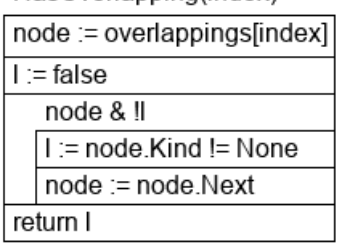
49. ábra



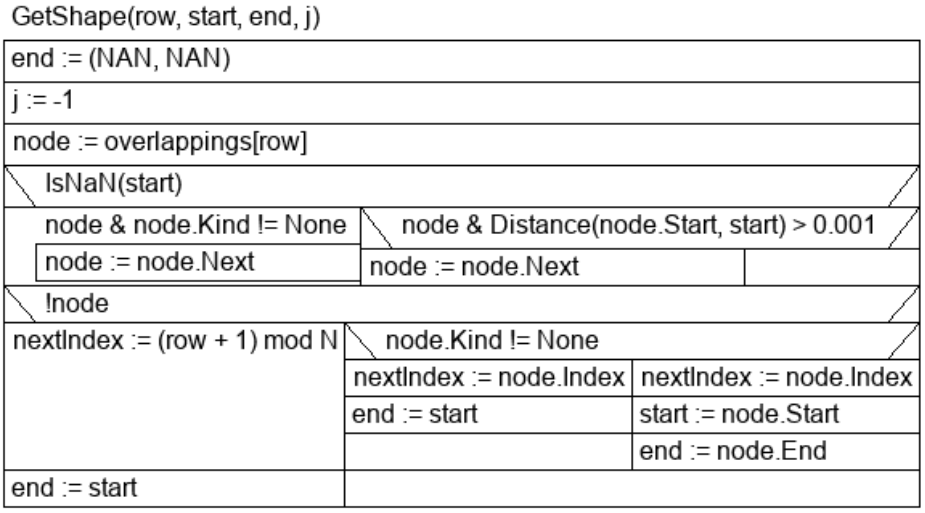
50. ábra



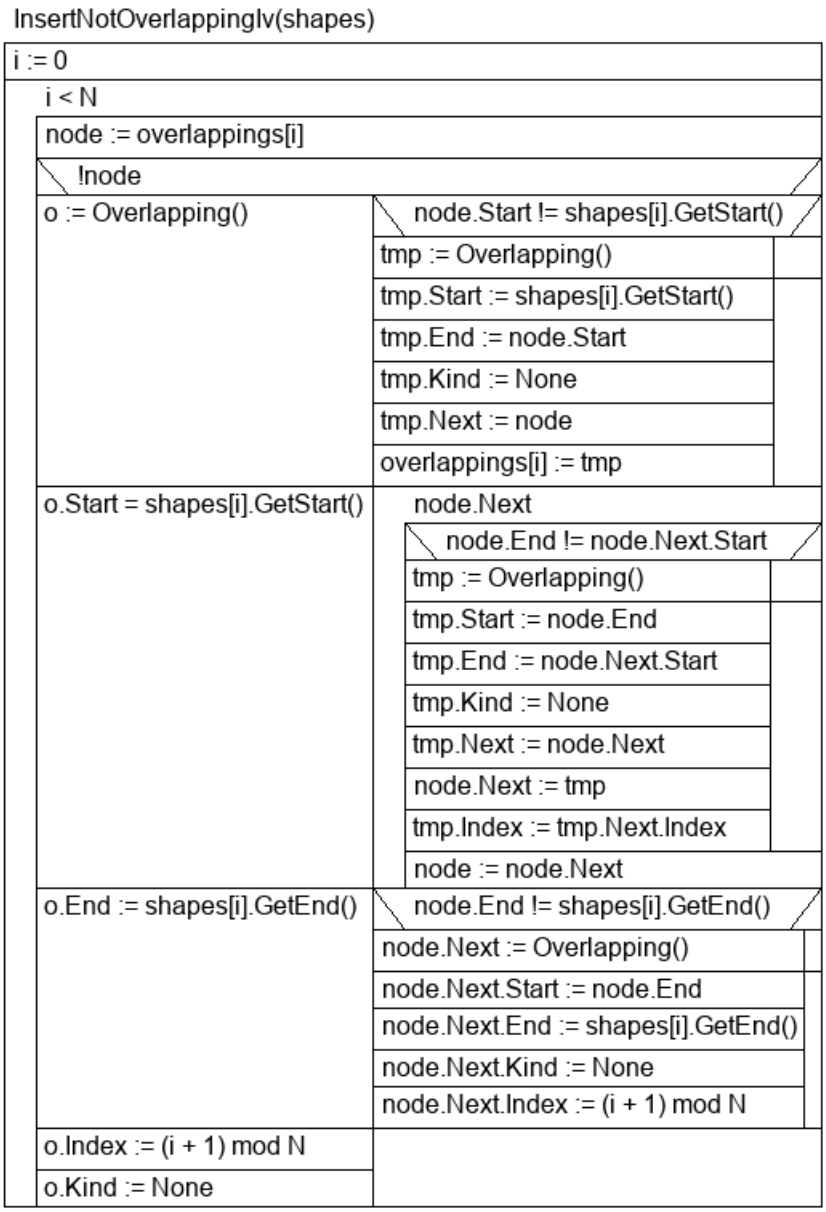
51. ábra



52. ábra



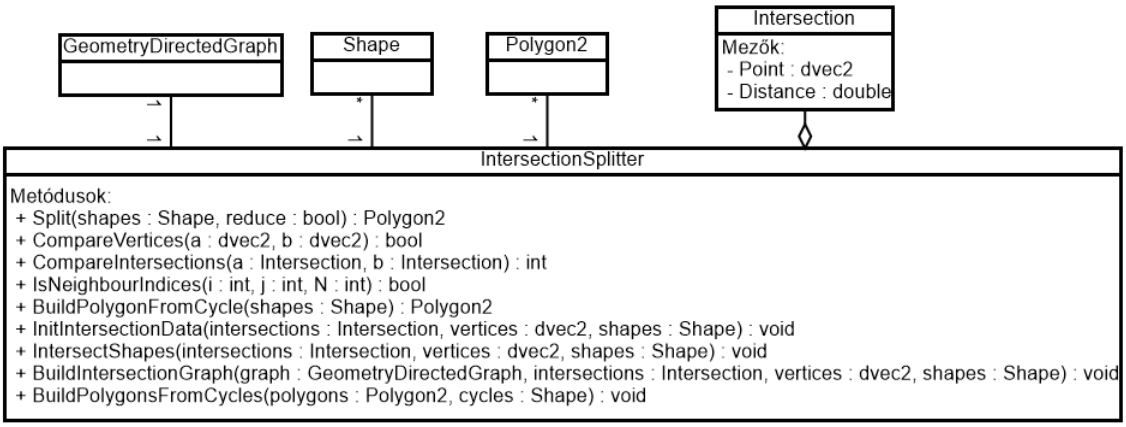
53. ábra


InsertNotOverlappingIv(shapes)
i := 0
i < N
node := overlappings[i]
!node
o := Overlapping()
node.Start != shapes[i].GetStart()
tmp := Overlapping()
tmp.Start := shapes[i].GetStart()
tmp.End := node.Start
tmp.Kind := None
tmp.Next := node
overlappings[i] := tmp
o.Start = shapes[i].GetStart()
node.Next
node.End != node.Next.Start
tmp := Overlapping()
tmp.Start := node.End
tmp.End := node.Next.Start
tmp.Kind := None
tmp.Next := node.Next
node.Next := tmp
tmp.Index := tmp.Next.Index
node := node.Next
o.End := shapes[i].GetEnd()
node.End != shapes[i].GetEnd()
node.Next := Overlapping()
node.Next.Start := node.End
node.Next.End := shapes[i].GetEnd()
node.Next.Kind := None
node.Next.Index := (i + 1) mod N
o.Index := (i + 1) mod N
o.Kind := None

54. ábra

- IntersectionSplitter

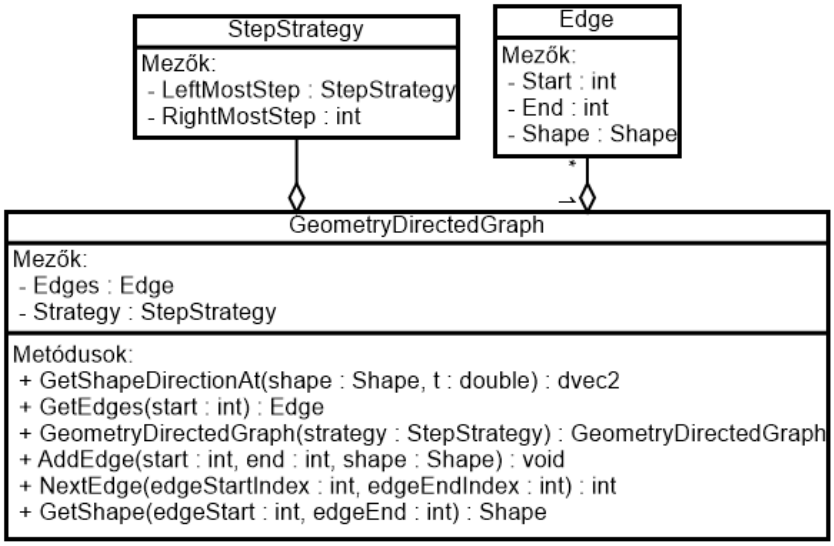
A poligon annak élkereszteződései alapján szétontásra szolgáló segéd-névtér. Ebben a névtétben több belső metódus található, amik a geometriai- illetve élkövetési gráf felépítésére szolgálnak, valamint a gráfban található körökből újabb poligonok létrehozására is. Maga ez algoritmus a „Split” publikus metóduson keresztül érhető el. Ennek bemenete egy éllista, valamint egy logikai érték, ami azt jelzi, hogy épp szűkítünk-e. Kimenete egy poligon lista, aminek minden elemére igaz az, hogy nincs benne egymást keresztező él, valamint a körbejárási iránya óramutató járásával ellentétes.



55. ábra

- GeometryDirectedGraph

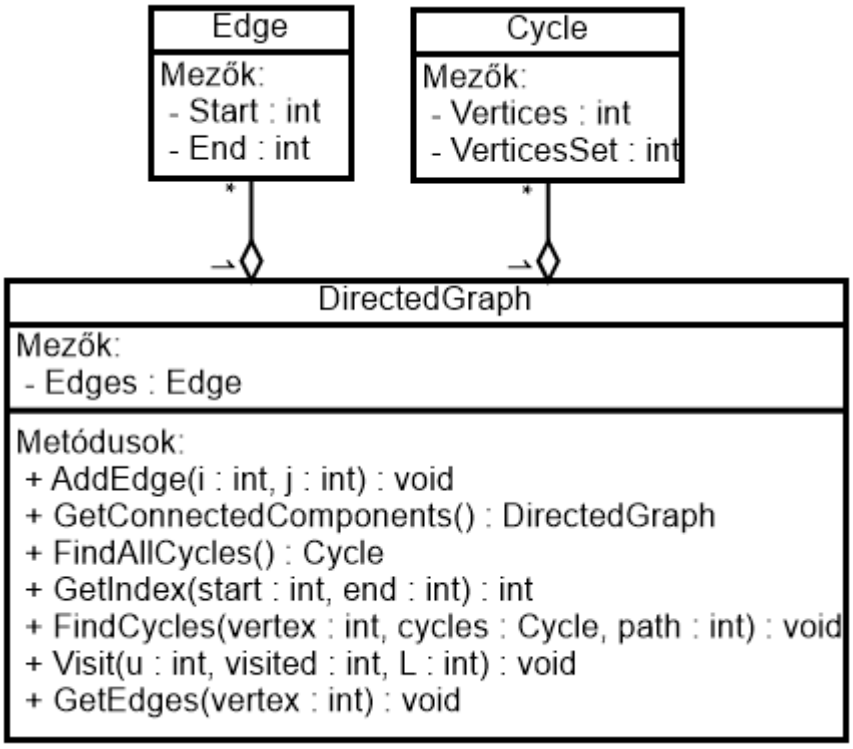
Olyan gráf tárolására szolgáló osztály, aminek a csúcsai és élei geometriai tartalommal is bírnak.



56. ábra

- DirectedGraph

Egyszerű irányított gráf tárolására szolgáló osztály. Implementál egy körkereső algoritmust, amivel le lehet kérdezni egy egyszerű gráfban lévő összes kört, valamint egy összetett gráfot egyszerű komponensekre bontó algoritmust.



57. ábra

3.2 Tesztelés

A következőkben néhány teszteset leírása és eredménye következik. A tesztesetek a „PolygonOffsetTest” projektben a „main.cpp” állományban találhatók.

Négyzet szűkítése egy tized egységgel	Egy darab poligon.
Négyzet bővítése egy egységgel	Egy darab poligon
Egység sugarú kör szűkítése egy tized egységgel	Egy darab poligon
Egység sugarú kör bővítése egy egységgel	Egy darab poligon
Fordított U alakú poligon szűkítése, a szűkítés után egymást fedő élek	Két darab poligon
Túlszűkített négyzet	Üres halmaz
Négyzet bármekkora nagy számmal való bővítése	Egy darab poligon

- Négyzet szűkítése egy tized egységgel

A teszteset célja a poligon szűkítés tesztelése. Egy négyzet szűkítése után pontosan egy poligonnak szabad maradnia az algoritmus végére. A bemeneti poligon megfelel az algoritmus előfeltételeinek.

Bemenet:

- Szakasz (1, 1) -> (-1, 1)
- Szakasz (-1, 1) -> (-1, -1)
- Szakasz (-1, -1) -> (1, -1)
- Szakasz (1, -1) -> (1, 1)

Kimenet:

- Szakasz (0.9 , 0.9) -> (-0.9, 0.9)
- Szakasz (-0.9, 0.9) -> (-0.9, -0.9)
- Szakasz (-0.9, -0.9) -> (0.9, -0.9)
- Szakasz (0.9, -0.9) -> (0.9, 0.9)

- Négyzet bővítése egy egységgel

A teszteset célja a poligon bővítés tesztelése. Egy négyzet bővítése után pontosan egy poligonnak szabad maradnia az algoritmus végére. A bemeneti poligon megfelel az algoritmus előfeltételeinek.

Bemenet:

- Szakasz (1, 1) -> (-1, 1)
- Szakasz (-1, 1) -> (-1, -1)
- Szakasz (-1, -1) -> (1, -1)
- Szakasz (1, -1) -> (1, 1)

Kimenet:

- Szakasz (2 , 2) -> (-2, 2)
- Szakasz (-2, 2) -> (-2, -2)
- Szakasz (-2, -2) -> (2, -2)
- Szakasz (2, -2) -> (2, 2)

- Egység sugarú kör szűkítése egy tized egységgel

A teszteset célja a poligon szűkítés tesztelése olyan poligonokra, ahol minimális, azaz két darab alakzat van. Az algoritmus eredményének egy darab poligonnak kell lennie. A bemeneti poligon megfelel az algoritmus előfeltételeinek.

Bemenet:

- Körív (1, 0) -> (-1, 0) origó középponttal
- Körív (-1, 0) -> (1, 0) origó középponttal

Kimenet:

- Körív (0.9 , 0) -> (-0.9, 0) origó középponttal
- Körív (-0.9, 0) -> (0.9, 0) origó középponttal

- Egység sugarú kör bővítése egy egységgel

A teszteset célja a poligon bővítés tesztelése olyan poligonokra, ahol minimális, azaz két darab alakzat van. Az algoritmus eredményének egy darab poligonnak kell lennie. A bemeneti poligon megfelel az algoritmus előfeltételeinek.

Bemenet:

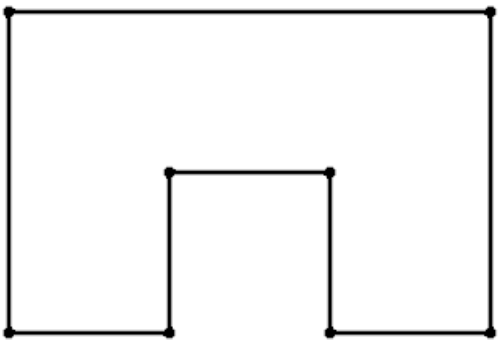
- Körív (1, 0) -> (-1, 0) origó középponttal
- Körív (-1, 0) -> (1, 0) origó középponttal

Kimenet:

- Körív (2 , 0) -> (-2, 0) origó középponttal
- Körív (-2, 0) -> (2, 0) origó középponttal

- Fordított U alakú poligon szűkítése, a szűkítés után egymást fedő élek

A teszteset célja az algoritmus vizsgálása olyan esetben, amikor a szűkítés során egymást fedő élek jönnek létre. A kimenet ebben az esetben két darab poligonnan kell lennie (59. ábra). A bemetet megfelel az algoritmus előfeltételeinek, amit az 58. ábra szemléltet.



58. ábra

59. ábra

- Túlszűkített négyzet

A teszteset célja olyan körülmény előidézése, amikor a szűkítés mértéke meghaladja a poligon méretét ezért az algoritmus eredménye egy üres halmaz. A tesztesetben egy négyzetre vizsgáljuk ezt a jelenséget.

Bemenet:

- Szakasz (1, 1) -> (-1, 1)
- Szakasz (-1, 1) -> (-1, -1)
- Szakasz (-1, -1) -> (1, -1)
- Szakasz (1, -1) -> (1, 1)

Kimenet:

- Üres halmaz

- Négyzet bármekkora nagy számmal való bővítése

A teszteset célja annak vizsgálata, hogy egy poligont bármilyen nagy, a számítógépen numerikusan ábrázolható számmal lehet bővíteni mindig lesz legalább egy darab poligon az eredményhalmazban. A mostani példában ezt egy négyzetre vizsgáljuk és a bővítés mértéke egymilliószoros.

Bemenet:

- Szakasz (1, 1) -> (-1, 1)
- Szakasz (-1, 1) -> (-1, -1)
- Szakasz (-1, -1) -> (1, -1)
- Szakasz (1, -1) -> (1, 1)

Kimenet:

- Szakasz (1 000 001 , 1 000 001) -> (-1 000 001, 1 000 001)
- Szakasz (-1 000 001, 1 000 001) -> (-1 000 001, -1 000 001)
- Szakasz (-1 000 001, -1 000 001) -> (1 000 001, -1 000 001)
- Szakasz (1 000 001, -1 000 001) -> (1 000 001, 1 000 001)

3.3 Továbbfejlesztési lehetőségek

A program számos irányba továbbfejleszthető, kiegészíthető új funkciókkal. A következőkben erre adok pár példát.

- Új típusú alakzat bevezetése

Ha a meglévő alakzattípusokon kívül (Szakasz, Körív) egy másikat szeretnénk bevezetni és használni a program lehetőséget add erre. Az új típus osztályának le kell származnia a „Shape” osztályból, implementálnia kell annak összes absztrakt metódusát. Az új típusnak el kell tudnia metszeni magát a már meglévő típusokkal, illetve egy egyenessel vett metszéspontjainak száma nem lehet több mint kettő.

Az új típus kezeléséhez szükséges a „Polygon2” osztály „Contains” metódusának módosítása is.

- Több vázlat kezelése

A felhasználó kényelme érdekében több vázlat betöltése és kezelése ugyan abban a programban célszerű megoldás lehet.

- Lyukas poligonok bevezetése

A program érzékelhetné, ha egy poligon egy másiknak a része, és ekkor úgy kezelné a belsőt, mint egy lyukat a külsőben. Szűkítéskor a külső szűkülne, a belső bővülne, bővítéskor fordítva. Az eredményt a külső és belső poligon metszete adná.

4. Irodalom

https://en.wikipedia.org/wiki/Point_in_polygon

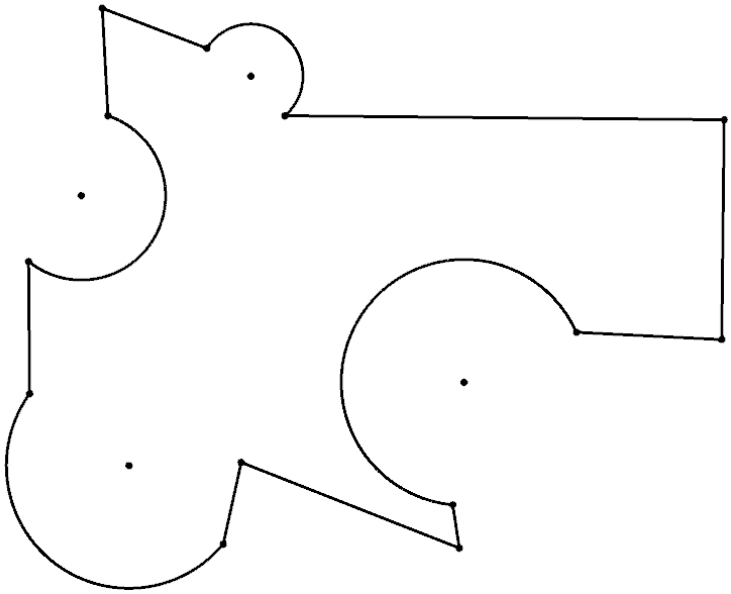
Elérés dátuma: 2019. május

https://mcmains.me.berkeley.edu/pubs/DAC05OffsetPolygon.pdf

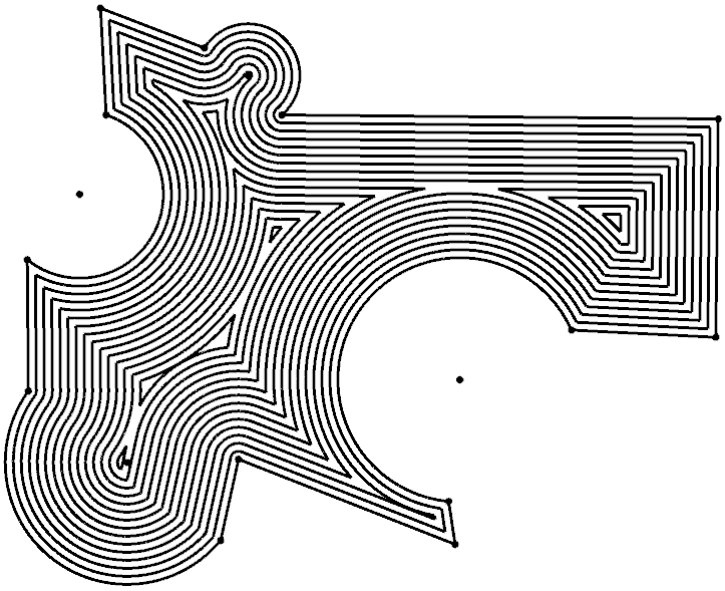
Elérés dátuma: 2019. május

5. Függelékek

5.1 Az algoritmus működés közben



60. ábra

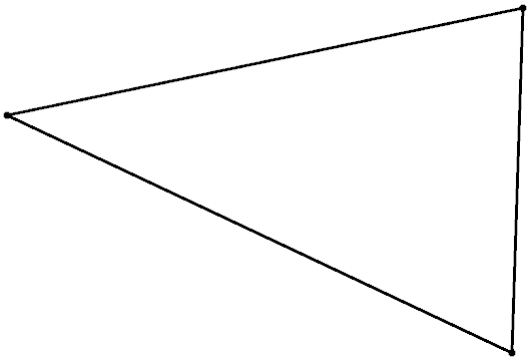


61. ábra

5.2. Példa fájlok

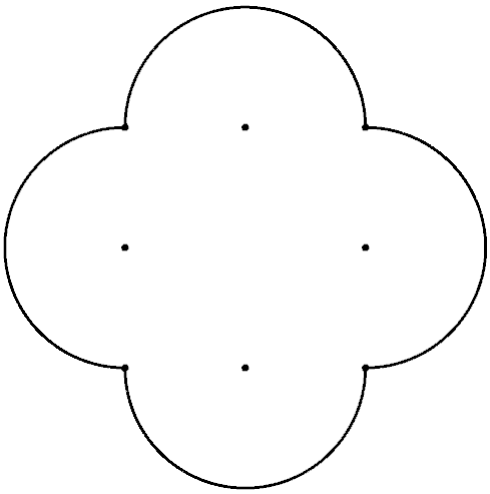
- triangle.txt

```
3
42.1;-43.9
93.6;-33.2
92.5;-67.6
3
Line 0;1
Line 0;2
Line 2;1
```



- arcs.txt

```
8
0;-20
-20;-20
20;-20
20;0
20;20
0;20
-20;20
-20;0
4
Arc 1;2;0;False
Arc 2;4;3;False
Arc 4;6;5;False
Arc 6;1;7;False
```



- random.txt

```
17
84.4;-61.6
98.3;-55.4
82.9967;-76.7552
58;-23.7
62.2;-28.6
52.5415;-20.2569
37;-38.5
40.3;-28.6
30.466;-46.6367
42.9;-71.9
30.6;-63
54.5722;-81.6087
83.8;-82.1
56.8;-71.5
39.6;-15.3
116.6;-29.1
116.3;-56.3
13
Arc 1;2;0;False
Arc 5;4;3;True
Arc 8;7;6;False
Arc 11;10;9;True
Line 2;12
Line 12;13
Line 13;11
Line 10;8
Line 7;14
Line 14;5
Line 4;15
Line 15;16
Line 16;1
```