Ádám Várhegyi-Miłoś

Y1JE9I@hallgato.uni-neumann.hu

Kivonat

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

Program Dokumentáció

Programozás 2 (levelező tagozat) – Beadandó feladat 16.

Tartalomjegyzék

[Tartalomjegyzék 1](#_Toc68560377)

[1. Fejlesztői dokumentáció 2](#_Toc68560378)

[1.1. A program célja 2](#_Toc68560379)

[1.2. Előzetes rendszerterv 2](#_Toc68560380)

[1.3. Fejlesztői környezet, programnyelv 4](#_Toc68560381)

[1.4. Részletes rendszerterv 4](#_Toc68560382)

[1.4.1. Képernyőtervek 4](#_Toc68560383)

[1.4.2. A forráskódban felhasznált változók 5](#_Toc68560384)

[1.4.3. Alkalmazott algoritmusok 11](#_Toc68560385)

[1.4.4. A program függvényeinek működése 11](#_Toc68560386)

[1.5. Tesztelés 11](#_Toc68560387)

[1.6. Továbbfejlesztési lehetőségek 11](#_Toc68560388)

[2. Felhasználói dokumentáció 12](#_Toc68560389)

[2.1. Futási környezet, követelmények 12](#_Toc68560390)

[2.2. A program használata 12](#_Toc68560391)

[2.3. Hibaüzenetek 12](#_Toc68560392)

[Ábrajegyzék 13](#_Toc68560393)

[Táblázatok jegyzéke 13](#_Toc68560394)

[Forrásjegyzék 14](#_Toc68560395)

# Fejlesztői dokumentáció

## A program célja

A program feladata egy bemeneti állomány beolvasása a program könyvtárából, abból egy lövészeti verseny adatainak kiolvasása kiértékelése, és az eredmények kiírása egy kimeneti állományba. Ezeken felül egy felhasználó által megadott versenyző adatait is részletesen ki kell elemeznie a programnak, és ennek eredményeit a képernyőre írnia.

Ez a program a Neumann János Egyetem GAMF Műszaki és Informatikai Karának Programozás 2 tantárgy (2021-es levelező kurzus) teljesítése céljából íródott, az ott leírt általános követelményeket és a szerző számára kiszabott feladatot igyekszik kielégítő módon megoldani.

## Előzetes rendszerterv

A bemeneti állománynak adott formátumúnak kell lennie a program működéséhez: az első sorban a versenyen részt vevők száma szerepeljen, majd minden további sor egy-egy versenyző lövéssorozatát jelenítse meg + és ­‒ karakterek segítségével, melyek közül előbbi a célt talált lövés, utóbbi pedig a célt tévesztett. A fájl neve „verseny.txt” legyen!

A megfelelő működés és a felhasználó által érthetetlen hibaüzenetek és elakadások elkerülése érdekében a program ellenőrizze a fenti feltételek teljesülését. A program generáljon hibaüzenetet és lépjen ki, amennyiben:

* az állomány megnyitása sikertelen,
* az állomány fejlécében nem szám szerepel,
* amennyiben szám szerepel az e szám nem egyezik meg a fájlban a szám alatt található sorok számával, vagy a sorok száma nem 2 és 100 közötti.

A programnak ezen felül úgy kell működnie, hogy amennyiben üres sorok szerepelnek a forrásfájlban a rekordok között, az ne okozzon problémát a fájl olvasásakor, és csak a nem üres sorokat vegye figyelembe a sorok megszámlálásakor éppúgy, mint a tartalom beolvasásakor.

Amennyiben az előzetes ellenőrzések sikeresek voltak, a programnak fel kell dolgoznia az állományban szereplő sorokat, melyekben kizárólag + és ­‒ karakterek szerepelhetnek. Ezt a programnak szintén érdemes ellenőriznie, és a felhasználó hibaüzenettel jeleznie, hogy érvénytelen karakter került az állományba, aminek feldolgozását így nem lehet minden kétséget kizárólag hibátlanul folytatni.

A + és ­‒ karakterek a fájlban a lövések eredményességét hivatottak jelölni: a pluszjel a célt talált lövést, míg a mínuszjel a célt tévesztett lövést jelenti. A programnak ezekből a jelekből meg kell tudnia állapítania, hogy a versenyzőknek:

* hány lövést adtak le,
* hány pontot szereztek.

A pontszerzést a következők szerint kell számolnia a programnak: a legelső lövéskor egy találat 20 pontot ér. Minden egyes célt tévesztéskor a következőleg megszerezhető pontok száma eggyel csökken, egészen addig, amíg el nem érjük a nullát. Fontos megjegyezni, hogy a versenyzők különböző számú lövéseket adnak le; nem adott számú alkalommal lőnek, hanem egy időkereten belül kell annyit lőniük, amennyit tudnak.

A beolvasott adatok alapján a program szóközökkel elválasztva kiírja azon versenyzők sorszámát, akiknek volt legalább kettő egymást követő találatuk, majd megadja azon versenyző sorszámát, aki a legtöbb lövést adta le (találattól vagy tévesztéstől függetlenül).

Az eddigieken felül a programnak egyéb információkat is ki kell számolnia egy versenyző lövéseiről, ezen versenyző rajtszámát a felhasználó a program futása során bekérésre adja meg:

* a versenyzőnek hányadik lövései találtak célt,
* összesen hány lövés talált célt,
* hány találatból állt a leghosszabb hibátlan sorozata,
* a versenyző pontszáma (amit már korábban kiszámoltunk).

Az előzetesen megfogalmazott igényeken túlmenően a program fejlesztője feltételezi, hogy a felhasználó esetlegesen további versenyzők adatait is szeretné lekérdezni, mely esetben a program újbóli és újbóli lefuttatása túlontúl körülményes volna. Ezért a programba beépítésre kerül egy egyszerű menürendszer, ahol az első versenyző részletes adatainak kiértékelése után a felhasználó meghatározhatja, hogy további versenyzők adatait is kívánja-e lekérdezni, vagy inkább a program további lépéseivel folytatná.

Utolsó lépésként a programnak el kell készítenie a verseny rangsorját az elért pontszámok alapján, melyet egy megadott nevű szöveges állományba kell exportálnia. Fontos megjegyezni, hogy rangsoroláskor kizárólag a pontszám veendő figyelembe; azonos pontszám esetén holtversenyben kapják a versenyzők a helyezést, majd utánuk kimarad egy helyezés (tehát két első helyezett esetén a második legnagyobb pontszámot elért versenyző a harmadik helyezést kapja, míg hármas holtverseny esetén a negyedik helyezés járna neki).

## Fejlesztői környezet, programnyelv

A program a Microsoft Visual Studio 2019 felhasználói környezetben kerül megírásra, C++ nyelven. Egyéb felhasználói környezet, illetve nyelv nem kerül felhasználásra.

## Részletes rendszerterv

### Képernyőtervek

Az 1.2 fejezetben leírtak alapján a bemeneti állománynak (mely a „verseny.txt” nevet kell viselnie) az 1. ábrán látható példa szerint kell kinéznie.

Eszerint a példa szerint a versenyen három résztvevő volt, lövéssorozataikat a plusz- és mínuszjelek jelzik. Külön indítóképernyővel a program nem rendelkezik, indításkor azonnal megnyitja, és beolvassa a forrásállományt.

1: példa a beolvasandó fájlra

3

+---++-

-+-+-+++

--+-++++--

Amennyiben a fájl nem hibás, a program elsőként kiírja a képernyőre a beolvasott adatokat, és a hozzájuk tartozó pontszámokat. Ezt azért teszi, hogy a felhasználónak legyen egy egységes képe a versenyzők eredményéről, illetve, hogy amikor egy-egy versenyzőre lekérdezi a részletesebb információkat, tudjon mi alapján választani a listából, és visszaellenőrizhesse a lekérdezés eredményeinek helyességét. A kiírandó lista az olvashatóság kedvéért táblázatszerű formába kerül rendezésre, ahogy azt a 2. ábra mutatja.

RAJTSZÁM LÖVÉSEK PONTSZÁM

1 +---++- 54

2 -+-+-+++ 88

3 --+-++++-- 86

2: a beolvasott adatok kiírása képernyőre

A legalább két egymást követő találatot jegyző, a legtöbb lövést leadó versenyzők sorszámai, majd a felhasználó által kiválasztott versenyző részletes adatai mind hasonló formában, tömören egy-egy sorban kerülnek kiírásra, előttük külön sorban a feladat sorszámával:

2. feladat:

Az egymast kovetoen tobbszor talalo versenyzok: 1 2 3

3. feladat:

A legtobb lovest leado versenyzo rajtszama: 3

Adjon meg egy rajtszamot! [ide írhatja a felhasználó a rajtszámot]

Amikor a program kiírta a felhasználó által kiválasztott versenyző részletes adatait, a felhasználónak lehetősége van egy újabb versenyző adatait lekérni, vagy tovább haladni a programmal. Ezt egy kezdetleges menürendszerrel oldja meg a program a 3. ábrán látható módon.

Az 1-es opció kiválasztásakor ismétlődik az előző lépés: a felhasználó újabb rajtszámot adhat meg, melyhez a program kiírja a vonatkozó adatokat, majd ismét előjön a fenti menürendszer. A 2-es opció kiválasztásával a program utolsó részéhez érkezünk.

3: opcióválasztó menü

\*\*\*\*\*\*\*\*\* VALASSZON EGY OPCIOT \*\*\*\*\*\*\*\*\*

1. Ujabb versenyzo adatainak lekerese
2. Kilepes a funkciobol es a program folytatasa

Az utolsó szekcióban, a program kiírja a képernyőre a teljes, immáron pontszámok alapján rangsorolt listát (lásd 4. ábra), és ezt a háttérben egy „rangsor.txt” állományba is kiexportálja (már a fejlécek nélkül).

4: a rangsor képernyőre írása

HELYEZES RAJTSZAM PONTSZAM

1 2 88

2 3 86

3 1 54

### A forráskódban felhasznált változók

#### A main függvény és kapcsolódó funkciói

A program main függvénye a loves.cpp forrásfájlban található. Itt kizárólag a program indítása, a lépések leírásának képernyőre írása, illetve adatbekérés történik, illetve a korábban említett menürendszert kezelő funkció. Az adatokat egy osztály dolgozza fel, ez külön forrásfájlban, illetve fejlécfájlban található, úgyhogy ezeket később részletezzük.

Ennek megfelelően a main függvényben csak a program indításához és a menürendszer kezeléséhez elengedhetetlenül szükséges változók vannak. Az bemeneti és kimeneti fájlok nevét egy-egy karaktertömb típusú változó tárolja, ezek az infajlnev és outfajlnev. A fájlt feldolgozó Loveszet osztályból a main függvényben egy L nevű objektum indít egy példányt. Továbbá clock\_t típusú kezd és veg nevű változók tárolják majd el a program futásának kezdetét és végét, melyek alapján a main függvényből való kilépés előtt ki tudjuk számolni a program futási idejét.

A main függvény kezeli 3. ábrán látható egyszerű menürendszert. A két opciót egy Opciok nevű enumerátor tartalmazza, a kapcsolódó int típusú menukiiras funkció végeredményét is ezen opciók egyikének felelteti meg a main függvény: ez a valasztas nevű integer. Mivel a menükiírást legalább egyszer futtatjuk, de bármennyiszer újra meghívhatjuk, egy while ciklusba van ágyazva, melyhez kapcsolódnak a bool típusú fkilep flag, és az int típusú bekeres.

táblázat 1: a loves.cpp forrásfájl változói

|  |  |  |  |
| --- | --- | --- | --- |
| Függvény | Változó neve | Változó típusa | Leírás |
| main | **infajlnev[]** | char | A bemeneti fájl nevét tárolja (kódba ágyazva). |
| main | **Outfajlnev[]** | char | A kimeneti fájl nevét tárolja (kódba ágyazva). |
| main | **L** | Loveszet | A Lövészet osztály egy példánya. Az ott kiszámolt adatokat az L publikus változói és metódusai írják a képernyőre. L-be írni a main függvény nem tud. |
| main | **kezd** | clock\_t | A program futásának kezdeti ideje. |
| main | **veg** | clock\_t | A program futásának vége. |
| main | **bekeres** | int | A felhasználó által megadott versenyző rajtszáma, input. A program ezt a rajtszámot adja meg a Loveszet ciklus azon metódusaiba argumentumként, amelyek egy-egy versenyző részletes adatainak lekérést szolgálják. |
| main | **Opciok** | enum | lekerdez = 1, kilep = 2  A menürendszer két opciója. A menukiiras int funkció kimenete a lenti valasztas változóban tárolódik el, ezt felelteti meg a main függvény valamelyik opciónak (vagy default-nak, ha a felhasználó érvénytelen opciót, pl. 3-at adott meg). |
| main | **valasztas** | int | A menukiiras funkció végeredménye kerül ebben a változóban eltárolásra. A main függvény ezt felelteti meg az Opciok enumerátor valamelyik elemével (vagy a default paraméterrel, ha a felhasználó érvénytelen opciót, pl. 3-at adott meg). |
| main | **fkilep** | bool | Szabályozza, hogy hányszor fussanak le a versenyzők részletes eredményeit kiíró osztálymetódusok és -funkciók, illetve a menürendszer. Értéke igaz, ha a felhasználó a menürendszerben a 2-es (kilép) opciót választja, vagy érvénytelen opciószámot ad meg. Ekkor a kilépünk a menürendszerből, és folytatódik a main függvény. |
| menukiiras | **opcio** | int | A kiválasztott menü opciója, ezt a változót adja vissza végül a függvény. |
| menukiiras | **cimhossz** | int | A menürendszer címének hossza. |
| menukiiras | **opcmaxhossz** | Int | A két opció leírása közül a hosszabbik szöveg hossza. A címhossz és ez a változó egyaránt felezve lesz, és ennyi számú szóközt ír a funkció a cím elé a középre zárás érdekében. |
| menukiiras | **cim** | string | A menürendszer címe. |
| menukiiras | **opcstr\_lekerdez** | string | A lekérdezési opció leírása. |
| menukiiras | **opcstr\_kilep** | string | A kilépési opció leírása. |
| menukiiras | **szokozok** | string | A cím középre zárásához meghatározott számú szóközre van szükség. Ez az *n* darab szóköz ebben a változóban tárolódik el, és a cím elé kerül képernyőre íráskor. |

#### A Lövészet osztály változói

A program központi eleme, a fájlkezelő változók, funkciók és eljárások, tehát a forrásállományt feldolgozó rész egy Loveszet nevű osztályban kerül megvalósításra. Az osztály kizárólag a fájlnevet kéri be argumentumként, az összes eddig felsorolt számítás az osztályban elhelyezett metódusokkal és eljárásokkal történik.

Az osztályban kizárólag azon változók, illetve függvények és eljárások publikusak, melyekhez valamilyen módon hozzá kell férnie a main függvénynek, mint például a létszám, pontszám, vagy a többi azon függvény, melyeknél különböző versenyzőkhöz kérhet le a felhasználó egyesével részletes adatokat.

Az versenyzők adatai (lövések, lövések száma, pontszám, helyezés, rajtszám) egy struktúra tömbben tárolódnak. Az osztály közvetlenül olyan adatokat tárol, melyek az egész versenyre érvényesek (pl. létszám). A versenyzők kerülhettek volna egy külön osztályba, és a Loveszet osztály mindegyik versenyzőre meghívhatott volna egy új példányt, de a programnak ebben a verziójára erre nem került sor, az egyszerűség kedvéért struktúra tömbben menti el a program a versenyzők adatait. Teszi ezt azért is, mert külön függvényekre, eljárásokra nincsen szükségünk a versenyzők esetében, kizárólag adatot teszünk bele, melyeket később kiolvasunk. Ez persze nem teljesen így van, hiszen a Loveszet osztálynak vannak metódusai és eljárásai, melyek egyszerre egyetlen, argumentumként megadott versenyzőre számolnak ki adatokat (pl. leghosszabb pozitív sorozat), de ezt ugyanígy meg lehet oldani a Loveszet osztályból is.

Fontos információ továbbá, hogy nincsen a Loveszet osztálynak olyan metódusa vagy eljárása, amely külső forrásból változtatná meg bármely privát változójának értékét. Ez azért van így, mert a verseny már megtörtént, az ott felvett adatok fixek, utólag nem változtatjuk meg őket; kizárólag a helyezéseket, statisztikákat állítja elő a program.

táblázat 2: a Lövészet osztály változói, eljárásai és metódusai

|  |  |  |  |
| --- | --- | --- | --- |
| Elérés | Változó neve | Változó típusa | Leírás |
| publikus | **Versenyzo** | struct | A versenyzők adatait táruló struktúra.  string slovesek, char clovesek[40]: A lövéssorozat +/‒ karaktereit tárolják.  Int pontszam: a versenyző pontszáma.  Int l: a lövések száma.  Int helyezes: a versenyző helyezése.  Int rajtszam: a versenyző rajtszáma. |
| publikus | **Loveszet** |  | A Loveszet osztály konstruktorja. Beolvassa a fájlt, megvizsgálja az érvényességét és meghív más függvényeket a pontszámok kiszámolásához és a rangsor elkészítéséhez |
| publikus | **~Loveszet** |  | A Loveszet osztály destruktorja. Törli a tmb és a rangsor struktúratömböket. |
| publikus | **getLetszam** | int | A verseny résztvevőinek számát adja vissza a main függvény számára. A main függvényben a felhasználótól bekérünk egy számot a részletes lekérdezéshez, ehhez kell tudnia a main függvénynek, hogy érvényes-e a felhasználó által megadott érték. |
| publikus | **getPontszam** | int | A felhasználó által bekért rajtszámhoz képest eggyel kevesebb indexű versenyző pontszámát adja vissza a main függvény számára. |
| publikus | **kiir** | void | A beolvasott forrásállomány tartalmát írja ki a képernyőre. |
| publikus | **adatexport** | void | A pontszámok alapján rangsorolt Versenyzo struktúratömböt tartalmát írja ki a képernyőre, és exportálja egy fájlba. |
| publikus | **min\_ket\_**  **talalat\_**  **eldontessel** | string | Eldöntés tételével megvizsgálja a Versenyzo tömb elemeit, és visszaadja azon versenyzők rajtszámait, akiknek volt legalább két egymást követő találatuk. |
| publikus | **min\_ket\_**  **talalat\_**  **stringfinddal** | string | Stringek find funkciójával megvizsgálja, mely versenyzők rekordjában szerepel a „++” szöveg, és ezen rajtszámokat visszaadja. Megegyezik a min\_ket\_talalat\_eldontessel metódussal, csak más megközelítéssel dolgozik. |
| publikus | **talalatok\_**  **sorszamai** | string | Egybefűzött szövegként visszaadja egy adott rajtszámú versenyző találatainak sorszámait (hányadik lövéseik találtak). |
| publikus | **legtobb\_loves** | int | Visszaadja a legtöbb lövést leadó versenyző sorszámát |
| publikus | **loertek** | int | Adott versenyzőre visszaadja a lövéssorozat alapján kiszámolt pontszámot. |
| publikus | **ossztalalt** | int | Adott versenyző összes találatainak száma. |
| publikus | **leghosszabb\_**  **sorozat** | int | Adott versenyzőre visszaadja, hány találatból áll a leghosszabb pozitív sorozata. |
| privát | **\*tmb** | Versenyzo | Struktúra tömb, melyben a versenyzők adatait tároljuk. |
| privát | **\*rangsor** | Versenyzo | Pontszámok alapján rangsorolt tmb struktúratömböt tárolja. |
| privát | **v** | int | A versenyzők létszáma. For ciklusokban releváns. |
| privát | **LOVES\_JELEK** | const string | +‒ jeleket tárolja a lövéseket leíró szövegek érvényességének megvizsgálásához. |
| privát | **PLUSZ** | const char | A + jel karakterkódját tárolja. |
| privát | **MINUSZ** | const char | A ‒ jel karakterkódját tárolja. |
| privát | **ervenyes\_szoveg** | bool | A forrásállomány adott sorára megmondja, érvényes-e, kizárólag érvényes karaktereket tartalmaz-e. |
| privát | **rangsorol** | void | Pontszámok alapján sorba rendezi a tmb struktúra tömböt, majd helyezéseket rendel a versenyzőkhöz (nem mindig egyezik a sorrenddel, mivel lehet holtverseny). |

A Loveszet osztály definíciója külön fájlban található a metódusaitól és eljárásaitól, erre azok viszonylag nagy száma miatt van szükség. A definíció a Loveszet.h fejlécállományban, míg az eljárások és metódusok a Loveszet.cpp állományban találhatóak.

táblázat 3: A Loveszet.cpp változói

|  |  |  |  |
| --- | --- | --- | --- |
| Függvény | Változó neve | Változó típusa | Leírás |
| Loveszet | **\*filenev** | char | Argumentum.  A beolvasandó bemeneti fájl nevét tárolja. |
| Loveszet | **be** | ifstream | A beolvasandó fájl objektuma. |
| Loveszet | **sorok\_szama** | unsigned int | A fájlban lévő sorok száma. Nem lehet negatív szám. |
| Loveszet | **fejlec\_letszam** | unsigned int | A beolvasandó fájl első sorában szereplő létszám. Nem lehet negatív szám. |
| Loveszet | **sor** | string | A beolvasott állomány egy-egy sorát tárolja el ideiglenesen a sorok megszámlálása céljából. |
| Loveszet | **nemerv\_index** | int | Az érvénytelen sorok számát menti el ide a program. |
| Loveszet | **i** | int | A sorok tartalmának beolvasásához és a tömbök elemszámához használt index. |
| ervenyes\_  szoveg | **&szoveg** | const string | Argumentum.  Adott sorban szereplő lövések karakterei. |
| ervenyes\_  szoveg | **erv\_karakterek** | string | Argumentum.  Elfogadható, a kiértékelés szempontjából érvényes karakterekből álló string. |
| ervenyes\_  szoveg | **rosszchar** | bool | Értéke igazra változik, ha érvénytelen karaktert talál a függvény a lövések rekordjában. |
| ervenyes\_  szoveg | **talalat** | int | A lövés sorozatok karaktereit egyesével megkeresi a program az érvényes plusz és mínusz karakterek alkotta stringben, ennek a keresésnek az eredménye kerül a talalat változóba. Ha az nulla, a karakter (és így a sor) érvénytelen. |
| ervenyes\_  szoveg | **\*szoveg** | char | Argumentum.  Amennyiben a lövések karaktersorozatként kerülnek beolvasásra, ez a mutató típusú változó tárolja a sorozat nulladik elemének memóriacímét. |
| ervenyes\_  szoveg | **h** | int | Argumentum.  Amennyiben a lövések karaktersorozatként kerülnek beolvasásra, ez a változó tartalmazza a sorozat hosszát. |
| ervenyes\_  szoveg | **i** | int | A lövés sorozatok karakterein végig haladó index. |
| min\_ket\_  talalat\_  eldontessel | **s** | string | Az eldöntés tételével azonosított, sorozatban legalább két találattal rendelkező versenyzők sorszámai kerülnek szóközzel elválasztva ebbe a stringbe. |
| min\_ket\_  talalat\_  eldontessel | **j** | int | Adott versenyző lövéseinek indexe. |
| min\_ket\_  talalat\_  stringfinddal | **s** | string | A lövés sorozatokat stringként beolvasva a kód mindegyik versenyzőnél egyenként eldönti, van-e legalább két találat. Az ilyen versenyzők sorszámai kerülnek szóközzel elválasztva ebbe a stringbe. |
| legtobb\_loves | **maxlovo** | int | A legtöbb lövést leadó versenyző indexe. |
| loertek | **\*sor** | char | Argumentum.  A lövés sorozatot alkotó karaktertömb nulladik elemének memóriacíme. |
| loertek | **hossz** | int | Argumentum.  A lövés sorozatot alkotó karaktertömb hossza. |
| loertek | **aktpont** | int | Az aktuálisan megszerezhető pontot tárolja egy cikluson belül; értéke minden egyes célt tévesztett lövésnél csökken. |
| loertek | **ertek** | int | Adott versenyző által megszerzett pontok számát tárolja egy cikluson belül; értéke minden egyes találatnál növekszik. |
|  | **rajtszam** | int | Argumentum.  Több metódusban is előfordul. A felhasználó által megadott lekérdezendő rajtszámot adja meg. |
|  | **r** | int | Több metódusban is előfordul. A felhasználó által megadott rajtszám argumentumként kerül be, r-nek eggyel kisebb értéket ad a program a nullával kezdődő indexálás miatt. |
| ossztalalt | **talalt** | int | Adott versenyző összes találatainak száma. |
| talalatok\_  sorszamai | **s** | string | A funkció visszaadja, adott versenyzőnek hányadik lövései találtak célt. Ezek a számok ebben a stringben, szóközzel elválasztva tárolódnak. |
| leghosszabb\_  sorozat | **talalt** | int | Adott versenyző lövésein végig haladva ebben a változóban tárolja el a kód az első lövés vagy a legutóbb tévesztés óta bevitt találatok számát. |
| leghosszabb\_  sorozat | **max** | int | Adott versenyző leghosszabb pozitív sorozatának elemszámát tárolja. |
| rangsorol | **\*a** | Versenyzo | Argumentum.  Az eredeti versenyző struktúra pontos mására megkreált rangsor tömbre mutató változó. Az eljárás ezt rendezi sorba. |
| rangsorol | **seged** | Versenyzo | Segédváltozó a sorba rendezéshez, két elem megcserélésekor használatos. |
| rangsorol | **szum** | int | Sorba rendezés után az eljárás helyezéseket rendel a versenyzőkhöz. A holtverseny miatt ez nem egyértelmű. Ez a változó számolja meg, hogy az aktuális versenyzőnél összesen hány versenyző szerzett több pontot. |
| adatexport | **ki** | ofstream | Az exportálandó állomány objektuma. |

### Alkalmazott algoritmusok

Asd

### A program függvényeinek működése

Asd

## Tesztelés

Asd

## Továbbfejlesztési lehetőségek

# Felhasználói dokumentáció

Asd

## Futási környezet, követelmények

Asd

## A program használata

Asd

## Hibaüzenetek

Asd

Ábrajegyzék

[1: példa a beolvasandó fájlra 4](file:///C:\Users\Adam\source\repos\prog2_lev_beadando16_y1je9i\prog2_lev16_y1je9i_dokumentacio.docx#_Toc68555411)

[2: a beolvasott adatok kiírása képernyőre 4](file:///C:\Users\Adam\source\repos\prog2_lev_beadando16_y1je9i\prog2_lev16_y1je9i_dokumentacio.docx#_Toc68555412)

[3: opcióválasztó menü 5](file:///C:\Users\Adam\source\repos\prog2_lev_beadando16_y1je9i\prog2_lev16_y1je9i_dokumentacio.docx#_Toc68555413)

[4: a rangsor képernyőre írása 5](file:///C:\Users\Adam\source\repos\prog2_lev_beadando16_y1je9i\prog2_lev16_y1je9i_dokumentacio.docx#_Toc68555414)

Táblázatok jegyzéke

[táblázat 1: a loves.cpp forrásfájl változói 6](#_Toc68555386)

[táblázat 2: a Lövészet osztály változói, eljárásai és metódusai 7](#_Toc68555387)

Forrásjegyzék

**There are no sources in the current document.**