Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

Kovács Boldizsár

ÖNÁLLÓ LABORATÓRIUM 2

RÉSZVÉNYADATOK STATISZTIKAI ELEMZÉSE ÉS VIZUALIZÁCIÓJA

KONZULENS

Dr. Goldschmidt Balázs

BUDAPEST, 2023

Tartalomjegyzék

Összefoglaló

Az önálló laboratórium során egy olyan programon dolgoztam, amivel a befektéseknél fontos kijelentést lehet megvizsgálni, hogy a múltbéli adatokból következtetni és azokra alapozni nem ad biztos befektetést önmagában. Ezt az állítást a részvények értékének alakulására vizsgálom, mint egy befektetési lehetőséget

A projekt során felmerült hálózati kommunikáció az adatok letöltésénél, a letöltött adatok eltárolása során adatfeldolgozás és adattárolás, fájlkezelés merült fel megoldandó problémának. A letöltött adatokat később újra be kell olvasni a részvények vizsgálatánál, elemzésénél, amely során mindent a memóriában nem lehetett eltárolni, így a fájlból beolvasást tettem hatékonyabbá, minthogy a háttérben való betöltést. Grafikus és tervezési szempontból a menürendszer felépítése és a saját interfész megvalósítása is a projekt részét képezte. Végül az elemzésnél a beolvasáson túl a lekérdezések előkészítését és megvalósításra kellett adnom eszközöket a felhasználó kezébe.

A projekt során részvényekkel kapcsolatos ismereteket gyűjtöttem előadások meghallgatásával, könyvek és online cikkek olvasásával, de a probléma komplexitása rajtam informatikusként túlmutat. A célom az volt, hogy sajátos igényekkel könnyen bővíthető programot alkossak, ami egyedi feltételek bővítésével szakemberek kezében hasznos eszköz legyen.

Törekedtem a program kódjának rendezettségére, használt algoritmusok hatékony megvalósítására, az interfész letisztult megalkotására és a részvények minél régebbi adatainak is a begyűjtéséről.

1 Bevezetés

Befektetési tanácsokat átadó előadások alapjait az képezi, hogy az előadó olyan ismereteket adjon át, amelyekkel a hallgatókat felvilágosítja a befektetésekkel kapcsolatban. Ha valamibe be kívánnak a jövőben fektetni, akkor tudjanak több megvizsgálandó tényezőről még a befektetés előtt és a felmerülő kockázatokról. Ezen előadásokon gyakran felmerülnek a kockázatok ismertetése mellet a különböző befektetési lehetőségek is.

Néhány ilyen előadáson részt vettem, és a részvényekről szóló részben mindig kitértek arra, hogy a részvényeket lehet technikailag elemezni, azaz matematikai alapokkal a múltbéli adatok alapján jellemezni lehet a részvények jelenlegi állapotát. [1] Valóban van több ilyen elemzési módszer, melyeket akár mutatókba szerveznek. Létezik például mozgó átlag, sztochasztikus vagy momentum elemzés részvények esetében és vannak indikátorok vagy mutatók, mint a MACD, RSI, ROC többek között, amelyek komplexebb számítással lehet megkapni és az általuk adott értéket valamilyen kategóriába sorolják be általában. Például az oszcillálás vizsgálatánál, azaz a görbe hullámzó mozgását vizsgálva kapott értéket úgy lehet kategorizálni, hogy éppen a hullám felső, alsó vagy köztes pontján tart a részvény.

Utóbbi példával élve a részvényt egy befektetőnek akkor érdemes vennie, amikor épp a hullám mélypontján van, mert akkor fog növekedni az értéke. Ilyen, és hasonló állításokat próbálnak a technikai elemzésekkel kijelenteni az elemzők. Viszont az előadások mindegyikében szintén kitértek arra, hogy ezen technikai elemzések önmagukban kevesek, és a múltbéli adatok alapján kiindulni nem ad biztos alapot a befektetéseknek. Fontos az alapvető elemzés is mellé. Alapvető elemzésnél a cég megítélése, munkájának értéke, az előállított termékének vagy szolgáltatásának minősége, maguk az alkalmazottak, a cég különböző szempontok alapján becsült értéke kerül előtérbe és ezek változása napról napra, negyedévről negyedévre.

A projektem célja, hogy írjak egy programot, amivel lehetőség nyílik megvizsgálni saját kezűleg azt, hogy csak múltbéli adatok alapján befektetni nem ad biztos(abb), jövedelmező alapot. Ehhez több száz részvényt kellene több szempontból a programmal megvizsgálni. Ezen program megtervezését és felépítését vázolom a következő részekben.

1.1 Program fő feladatai

Az elemzés során szükség lehet a szempontokkal, feltételekkel kísérletezni, megvizsgálni, hogy mely mutatók, indikátorok, értékek, feltételek vagy tények 8továbbá címkék) akár együttes értéke, megléte ad tényleges befektetési alapot. Ezért kell egy adatbázis lekérdezéshez hasonló információ kinyerésére alkalmas része a programnak a részvények múltbéli adataiból. Ezen lekérdezéseket elő kell készíteni, a feltételeket, címkéket létre kell hozni és az eredményt megjeleníteni, vagy fájlba kiírni.

A lényegi másik, pontosabban első része a programnak a lekérdezéseknél használt adatok összegyűjtése, kezelése, tárolása és előfeldolgozása. A részvények múltbéli adatait le kell tölteni valahonnan, ha nincs közvetlen hozzáférése valakinek régi adatokhoz fizikai adathordozón. Ekkor kell keresni egy megfelelő adatszolgáltatót, amely képes megfelelő adatokat szolgáltatni. Az feltétel vele kapcsolatban az, hogy lehetőleg minél régebbről, minél pontosabb, teljesebb és hibátlan adatokat tudjon szolgáltatni perc bontásban. Fontos, hogy minden nap minden percéről legyen rekord a részvények értékének alakulásáról, hogy lehetőség legyen részletesebb elemzésre.

A letöltött adatok ennek ellenére lehetnek hibásak vagy ritkák. Ritka például, ha egy napról összesen csak két percről van információ. Hibás, ha az időpont, vagy a részvény értéke nem lehetséges értéket vesz fel. Ezen hibákat javítani kell a letöltést követően, vagy a legutolsó letöltést meg kell ismételni helyes adatok reményében. A letöltött, majd javított adatok a háttértáron kerülnek eltárolásra.

A több száz részvény akár húsz évre visszamenő percbontásban vett értéke sok adatot jelent, legalábbis ennyi adatot eltárolni memóriában egy személyi számítógép esetében ritka, a háztartásokban használt gépeken gyakran nem lehetséges. Jelenleg a letöltött adatok (nem tömörítve) több, mint 60 GiB-et foglalnak. Ennyi adatot a memóriába betölteni egyszerre nem tudok, így más megoldást kerestem az adatok feldolgozására.

Fontos a letöltésnél, hogy mely részvényeket töltse le, ahogy a lekérdezéseknél is, hogy mely részvényekre futtassa le a lekérdezést. Ehhez össze kell gyűjteni az érdekes részvényeket, és csoportokba szervezni, hogy mind a lekérdezés, mind az adatok frissítése lehetséges legyen csak adott részvény csoportokra.

Már van adat, csak a lekérdezéshez a címkék hiányoznak. Ehhez paraméterezhető címkék kellenek és a részvények programon belüli képe. A részvények értékének alakulását megjeleníteni lehetővé teszi a szabad szemmel történő elemzést és a címkék létrehozásánál közvetlenül ott lehet a létrehozás során a minta, ami alapján a címke paraméterezve lesz.

1.2 Grafikus megjelenés

A program kezelése könnyebb, ha van grafikus interfésze, továbbá a részvények értékének és elemzésének vizualizációjában nagyban segít. A programomat C++ -ban írom, a Windows Froms .NET-es megoldást így nem használom. Azaz a menürendszert és a részvények görbéjét kirajzolni, a bemeneteket és a kimeneteket kezelni saját megoldásommal teszem. A grafikus és inputkezelő könyvtárnak az SDL2-t [2] használom. A program használatához a billentyűzetet és az egeret egyaránt kell használni.

2 Adatok begyűjtése és tárolása

2.1 Adatok letöltése

C++ -ban a CURL külső könyvtárral oldottam meg az API hívások összeállítását, küldését és az adatok fogadását is. Viszonylag egyszerűen használható könyvtár, de a biztonsági protokollokat nem kezelte első körben jól, viszont erre talán nincs szükség ezen programnak, így nincs probléma vele.

Az adatszolgáltató honlapnak a www.alphavantage.co választottam. Itt lehet a részvényeket több, mint húsz évre visszamenőleg perc bontásban letölteni. A letöltést API hívásokkal lehet megvalósítani. A hívásokat két paraméterrel kellett ellátnom, a letöltendő részvény rövidítésével és a kívánt év és adott hónapjával. A hívás konstans paramétere a perc bontású adatok kérése, és az adatok .csv fájlba és vesszővel elválasztott formába való letöltése volt. A részvény lista megléte esetén az adatok letöltésénél egy adott részvény csoport minden adatát letölti. Ez részvényenként a negyedéves jelentéseit, és havi tagolásban a perc bontású értékét jelenti.

Az adatszolgáltatónak van ingyenes verziója, így tudtam előre tesztelni, hogy milyen hibák léphetnek fel letöltés során. Felléphet hívás korlát hiba, hogy nincs több letöltési lehetőségem az adott percben, vagy napon, mert egy nap csak 100, egy percben csak 5 alkalommal tölthetek le adatokat (fizetős verzióval csak a percenkénti letöltésre van korlát, és abból adódó egész napra vett korlát). A másik gyakori hiba, hogy az adott részvény nem létezik az adatszolgáltatónál, vagy más rövidítés alatt, így nem tud adatot visszaadni. Ekkor a részvényt vagy egyáltalán nem lehet letölteni, vagy módosítani kell a rövidítését a listában. Ritka esetekben a szolgáltató eseti, szerver oldali hibát dob. Ezt szintén kezelnem kellett.

A letöltési korlátot kliensoldalon is limitálom, hogy feleslegesen ne próbálkozz a letölteni adatokat gyakrabban, illetve a visszaküldött hibaüzenetet felismerem, és várakozom a következő perig abban az eseteben. A letöltést külön szálon valósítottam meg, hogy ne blokkolja se a grafikus felületet, az interfészt. Ha nem létezik a részvény, akkor azt a részvény negyedéves jelentésének letöltésével ellenőrzöm. Ha nincs negyedéves jelentése, akkor az vagy nem annyira komoly cég, vagy ténylegesen nem létezik, esetleg nagyon új. Ekkor a részvény letöltését a listából kihagyom. Ezért kezdi a program a részvények negyedéves jelentéseinek letöltésével először, hogy kiszűrje a nem

létező részvényeket. Az eseti hibákat felismeri a program és a letöltéssel újra megpróbálkozik. Ez majdnem mindig megoldja a problémát. Ha valamiért nem oldja meg a hibát a többszöri letöltés, akkor tovább lép. Ilyen még nem történt 800+ részvény letöltése során.

A letöltött adatokat egy kijelölt mappában tárolom el. A részvényeket a gyökérmappán belül külön mappákban tárolom el. Egy részvény esetében a negyedéves jelentések két fájlban kerülnek eltárolásra, mert két API hívás szükséges hozzájuk, egy minden nap adatát (nyitó, minimum, maximum, záró és kereskedés volumen értékét) tartalmazó összesítő fájl, ahol az összes nap szerepel. Ez nem teljesen igaz, ugyanis az összefüggő napokat tartalmazza csak az elmúlt évekre visszamenőleg. Volt ugyanis olyan, hogy a havi részletekben letölthető perc bontású adatokból volt 2007-es, de ebben az összesítőben az legkorábbi nap 2018-as volt.

Ennek okán nem tudtam a havi részletekben letölthető adatokat az alapján letölteni, hogy melyik nap az első, és azon nap hónapjától kezdve töltöm le az adatokat, hanem végig próbálja a program letölteni a részvények adatát 2000 januártól. Gyakran a cégeknek nincs ilyen távra visszamenő tőzsdei múltja, így innen következett egy másik hiba, hogy a részvény létezik, van negyedéves jelentése, de nem lehet tudni, hogy melyik hónapokról van letölthető adat. Külön nem lehet lekérdezni ezt a szolgáltatótól, így csak végig próbálni lehetséges. A legrégebbi hónaptól nem szükséges, hogy folyton legyen adat, így a hónapok listája sem folytonos. A havi részletekben letöltött adatokat egy-egy fájlban tárolom el a részvény mappáján belül egy mappában.

2.2 Adatok feldolgozása

Ha nincs felmerülő hiba, akkor a letöltött adatot eltárolása előtt még fel kell dolgozni. Az adatokat vessző mentesítem és a hiányzó vagy hibás adatokat kipótolom, kijavítom. A hibás az adat, például egyetlen érték kiugróan magas, vagy 0 vagy hiányzik, akkor a legutolsó ismert értékkel felülírom, kipótolom. Ekkor az adatok beolvasásánál már nem kell figyelni esetleges hibákra, hanem elég csak minél hatékonyabban beolvasni őket. Minden nap 390 percen keresztül van nyitva a tőzsde, kivéve hétvégéken és ünnepnapokon. Amikor a tőzsdéről beszélek, akkor az amerikairól beszélek, a NASDAQ-ról vagy az NYSE-ről. A nyitvatartás magyar idő szerint nagyban változik, nyári és téli időszámítás miatt, de a letöltött adatok esetében szerencsére nem. Mindig 9 óra 30 perctől van nyitva a tőzsde, és 16 órakor zár. Emiatt az időt átalakítanom nem kellet, viszont a

nyitás előttről és zárást követően is lehetnek adatok. Ezeket jelenleg eldobom és csak a nyitás utáni 390 percre koncentrál a program. Van olyan eset, mikor van egy adott napról adat, viszont nincs a tőzsde nyitása idejéről adat, csak a nyitás előttről. Ekkor azzal az adattal töltöm fel az egész napot.

Ilyen adatok használata szerintem nem javasolt, de képzettebb elemzők lehet, hogy tudnak ebből is hasznos információt kinyerni, így ezzel a megoldással hagytam meg a programot. A havi adatokat nem fésültem össze egy fájlba, mert jelentős javulást nem értem el a beolvasásnál, és az adatokat elegendő így csak az adott intervallumon belüli fájlokból kinyerni, plusz a hozzájuk tartozó intervallumból. Például a 200 napos mozgóátlag esetén nem csak az adott hónap kell, hanem az azt megelőző 7 is.

2.3 Adatok kezelése

Az adatokat nem mindig semmi előélettel letölti újra, hanem ha már letöltötte egy részvény adatait 2000-től 2013 májusáig, akkor lehet a letöltést onnan folytatni, így az API hívás korlátot hatékonyabban használja ki. Ezt a megoldást az adatok frissítésének hívom. A frissítés során a részvény 3 gyökérmappában lévő negyedéves jelentések és a napokat összesítő fájlokat tölti le először, hiszen azok naponta változhatnak, a havi adatokat a legutolsó meglévő hónaptól kezdi el. Azért szükséges onnan kezdeni, hogy ha az a hónap csak 10-én volt legutoljára frissítve, akkor a hónap végi adatokat is letöltse korábban, minthogy tovább lépjen azon adatok nélkül.

Ezen megoldás csak a fájlok meglétének ellenőrzéséből és a letöltés kis módosításából állt, viszont nagyban meggyorsította az adatok letöltését a letöltési korlát miatt. A letöltéshez és frissítéshez egy töltő csíkot is kreáltam, hogy a letöltés vagy frissítés állapotáról képet kapjak, továbbá a várhatóan hátramaradó időre is becslét írjak. Utóbbival nehézségek támadta, ugyanis pontos képet nem tudtam adni, mivel a letöltés során 2000-től a mai napig 288 hónapot kell letölteni. Ebből, ha nem kell letölteni az első néhányak, akkor az letöltött hónap / az összes letölteni kívánt hónap törtből a nevezőt csökkenteni kell. Ekkor viszont nem kapunk reális képet arról, hogy mennyi időbe telik még befejeznie a letöltést, viszont biztos felső becsléssel szolgál. Sajnos ez 2 hétről indult a letöltési korlát miatt, de több részvénynek nincsen csak összesen 48 hónapról adata, emiatt a letöltés lerövidült 2-3 napra.

Egy kis javítást ejtettem, ezzel a letöltést azzal egészítettem ki, hogy az eddig letöltött hónapok / megpróbált hónapok arányát is számontartom. Így a hátramaradó

hónapokat ezzel az értékkel szorzom be, így egy becslést kapok arról, hogy várhatóan hány hónapot nem kell közölük letölteni, amivel közelebb került a valós időhöz a program, viszont nem ad biztos felső becslést innentől a program.

2.4 Adatok beolvasása

Az adatok beolvasása kritikus ekkora méretű adatnál. A háttértártól függően különböző megoldásaim különböző gyorsasággal adták vissza a adatokat és tudtam betölteni a memóriába. Először is az adatokat lehet a memóriába egyszerre betölteni, a „memory mapping”-et használva. Ekkor a fájlokat nem olvassa be soronként, még ha a felhasználó úgy is kívánja beolvasni a fájlokat, helyette a memóriába a háttértárról minden adatot előre betölt és a memóriából majd beolvassa az adatokat rendezett formába tároláshoz. Ezt lehetett több szálon is megvalósítani például egyszerre több részvényt párhuzamosan beolvasni ilyen módon. A háttértárak, amiken vizsgáltam a megoldásokat, viszont teljesen különböző módon reagáltak. A párhuzamosítás egyiken gyorsított, míg egy másikon csak és drasztikusan lassított. A memóriába előre betöltés [3] viszont negatív hatással nem volt a teljesítményre, inkább csak pozitívval.

Az adatok beolvasásánál két féle számtípust kell beolvasni. Egészeket, mint az év, hónap, nap, óra, perc és a kereskedés volumene, illetve törteket, a részvény értékeit. A tört számok beolvasása viszont lassú feladat. A formátumok különbözősége miatt, többféle módon is ki lehet írni ugyan annak beolvasott számot. Viszont itt tudtam, hogy pontosan milyen formátumban vannak az értékek, így saját történték beolvasást valósítottam meg, ami szintén gyorsított nagyban, olyan 60%-ban a beolvasás sebességén (nem csak törtek vannak, hanem egészek is, így összességben a 60% remek). [4]

Leglassabb háttértáron, amit találtam, egy memóriával alig rendelkező merevlemezről a beolvasás 4 MiB/másodperc sebességgel történik, azaz egy részvényt lehet, hogy akár 50 másodpercig is tölt. A leggyorsabb esetében ez 150+ MiB / másodperc lett, amivel a beolvasása egy részvénynek átlagosan fél másodpercre szorítottam vissza ezzel a jelenlegi 800+ letöltött részvény 65+ GiB mennyiségű adatát 7 és fél perc alatt mind beolvassa. Ennél gyorsabb megoldások is lehetségesek, de a fájlokat nem kívántam tömöríteni és beolvasásnál pluszban kitömöríteni. A fél másodperc egy részvényre viszont ad elegendő időt (lehet ez már kevés is) a lekérdezés elvégzésére is a memóriában, így ennél a megoldásnál maradtam.

2.5 Adatszerkezet

A részvények nevét egy string-ben, a napjait és a negyedéveit külön egy-egy set-ben eltárolja. Set felépítése elhanyagolható a fájlból beolvasáshoz képest, ugyanis csak 5800 elem lehet jelenleg egy set-ben.

Egy naphoz eltárolja az évet a hónapot és a napot, ami az azonosítójaként szolgál a set-ben, eltárolja hozzá, hogy az a hét melyik napja illetve az azt követő és az azt megelőző nap dátumát, a napi minimum, maximum és nyitó, illetve záró értékét, továbbá a percenkénti adatokat egy vektorban. A vektorban már indexszel kereshető az adott nap adott perce, így nem kell set. A dátumhoz azért szükséges viszont a set, mert nem elegendő eldönteni, hogy hányadik napról van szó például 2000. 01. 01-től, hanem az ünnepnapokat is számításba kell venni, továbbá, ha az első rekord későbbi, mint a millennium kezdete. A percenkénti adatoknál eltárolja az órát és a percet is a részvény adott értékei mellett, hogy ne kelljen folyton újra kiszámolni az indexüket, és önmaguk is tudják a helyüket a vektorban.

Egy negyedévhez eltárolja a negyedévi jelentés záró dátumát, a jelentés kiadásának tényleges dátumát, a jelentés korrigált dátumát, ami azért korrigált, hogy azon dátumot kapja meg, amelyik nap nyitása előtt jött ki a jelentés. A negyedéves pénzügyi jelentések általában a tőzsde nyitása előtt vagy a zárás után adják ki, emiatt, ha zárás után adják ki, akkor a korrigált dátum az a másnapi dátum lesz, mert az lesz az a nap, amelyik nyitása előtt közvetlenül jött ki a pénzügyi jelentés. Eltárolja a következő jelentés kiadását megelőző napot, azaz ennek a negyedévnek a végét és a negyedéves jelentés néhány kulcs értékét, mint az osztalékot, a bevételt és a profitot, illetve az ehhez tartozó elemzők becsült értékét és százalékos eltérését.

3 Menürendszer és az interfész

Az interfész teljesen saját, így minden egyes eleme saját. A menük a program indításakor mind létrejönnek és a menükben végzett folyamatok eltárolásra kerülnek így, mikor elvált a felhasználó akár másik menüre is, hiszen a háttérben létezik még a másik menü. A menüket (MenuK) eltároló osztály a program indítása során létrehozza a menüket és a menük közti navigációhoz a menüknek paraméterben átadja azoknak a menüknek a referenciáját, amelyeket abból a menüből el lehet érni közvetlenül. A többi menübe navigálás esetén az aktuális menü pointerét kell csak átállítani a másik menüre így. A menük felépítése és bemenet feldolgozása menünkként egyedi, mert a benne szereplő elemeket nem minden esetben használhatóak ugyan úgy, mint másik menükben.

3.1 Interfész elemek

Az egyszerűbb elemeket is meg kellett valósítanom, hiszen beépített megoldás nincsen hozzá az SDL-ben. Emiatt minden elemet magam megszabhatom, hogy miként viselkedjen milyen input esetén. Elemi műveletei a grafikus könyvtárnak a szöveg kiírása, ami egyetlen betűtípust és betűméretet használ, ezeken kívül geometriai elemeket lehet csak kirajzolni, mint a téglalapot, kört, vonalat különböző színekben és kitöltéssel. Lehet még képeket és pixel szinten kirajzolni vele, de erre nem volt szükség.

3.1.1 Gomb

Egyszerű téglalap, amit kibővítettem azzal, hogy opcionálisan kijelölhető is. Ekkor egy kis kör megjelenik a gombon belüli szöveg végén, és az zöldre vált, ha kijelöli. Esetenként a kijelölés hatására a gomb szövege is megváltozhat.

Valamikor menüt vált egy gomb megnyomására, valamikor egy listából kiveszi, vagy hozzáveszi az adott gombhoz tartozó értéket. Valamikor meg egy folyamatot indít el, egy függvény hív meg. A menünkként létrehozott gombok viselkedése kattintásra a menü bemeneténél vannak lekezelve.

3.1.2 Szöveg beviteli mező

Valamikor meg kell adni egy nevet, egy dátumot, egy fájl nevet, akkor használom ezt. Gyakran kiegészül egy görgethető listával, amiből a bevitt szöveg alapján szűr a megmaradt értékekre. Ezen beviteli mezőkben lévő szöveggel dolgozik néhány függvény.

3.1.3 Görgethető lista

A listába gombokat szervezek, amelyek vagy egy szöveg beviteli mezőhöz van kötve és így a gombot megnyomva a beviteli mezőt kitölti az adott szöveggel, vagy kijelölhető gombokból áll a lista, így végig lehet görgetni például a részvényeken, hogy egy részvénycsoportba melyik részvények tartoznak bele. A kijelölünk hozzájuk néhányat, vagy elveszünk közölük néhányat, akkor tudja módosítani a részvénycsoportokat. Ezek hossza miatt a nyilakkal lehet növelni a görgetés léptékét, amelyet a oldalsó csúszka színének elváltozása mutat.

3.1.4 Töltési csík

A töltési csíkot a hosszabb folyamatok állapotának kijelzésére hoztam létre. Lehet egy ilyen elemet elindítani. Ekkor a hozzá társított folyamatot elindítja. Lehet törölni a hozzá társított folyamatot és lehet az elindított folyamatot megszakítani.

A folyamat visszahívhatja három függvényét. Egyikkel az elméleti maximum lépést tudatja a töltő csíkkal a folyamat, a másodikkal jelzi, hogy végzett adott számú lépéssel, és a harmadikkal, hogy kihagyott néhány lépés. A töltő csík számontartja az elvégzett lépéseket és az összes lépést. Ezen értékeket módosítja az előző három függvény. Az eltelt idő és az említett értékek alapján kiszámol egy várható befejezéshez szükséges időt.

A csík színe változik a folyamat előrehaladtával és az előrehaladást százalékban is kifejezi, illetve, ha végzett a folyamat, akkor a csík színe nagyot változik és szöveggel is jelzi leállást.

3.1.5 Részvény

Egy részvény kirajzolásához használom az előzőek mindegyikét, továbbá a részvények értékeiből képzek egy görbét. A görbe értékeit gyertyákkal rajzoltatom ki. Lehetőség van kiválasztani megvizsgálni kívánt részvényt majd a részvény adott negyedévét vagy napját, és onnan előre és hátra lépni közöttük. Kiírja hozzá az előző dátumhoz képest a zárás és nyitás közötti eltérést százalékosan, a negyedév esetében a dátumokat, nap estében az időpontokat írja ki a vízszintes tengelyen, a függőlegesen a részvény konkrét értékét vagy a nyitáshoz, vagy az előző nap, illetve negyedév zárásához képest és igazítja hozzá a grafikont.

3.1.6 Új címke létrehozása

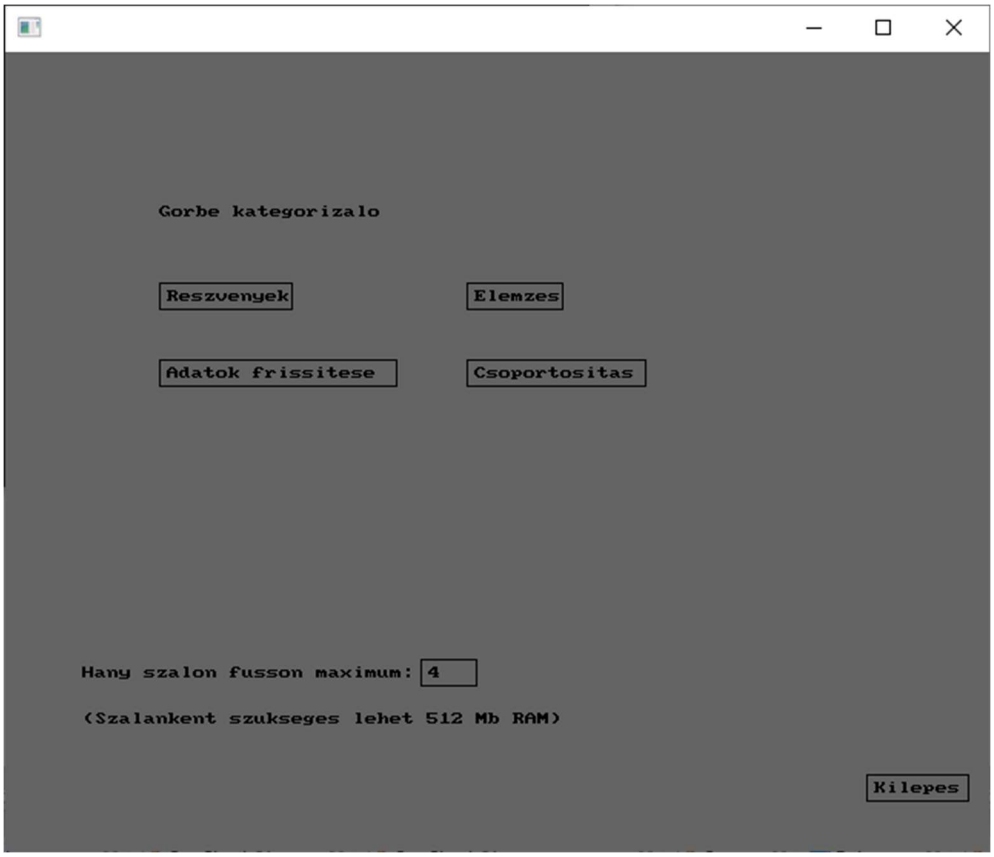
Az újabb címkék létrehozására a részvények megtekintésénél van lehetőség. Ott egy felugró ablak jellegű megoldást választottam. A címke típusát kiválasztva feldobja a címke paramétereit, amiket ki kell tölteni egy új címke létrehozásához. Itt van először dinamikus beviteli mező. Ezek inputmezőket a címke a létrehozáskor megkapja, feldolgozza és létrehozza az új címkét. Ha a megadott paraméterek hibásak, akkor eldobja a létrehozási kérést.

3.2 Menük

A menükben gombokkal át lehet navigálni másik menükre. Ezeket az adott menü szomszédos menüjeinek hívom. A leírt interfész elemekkel megépített menüket listázom funkcióikat részletezem.

3.2.1 Főmenü

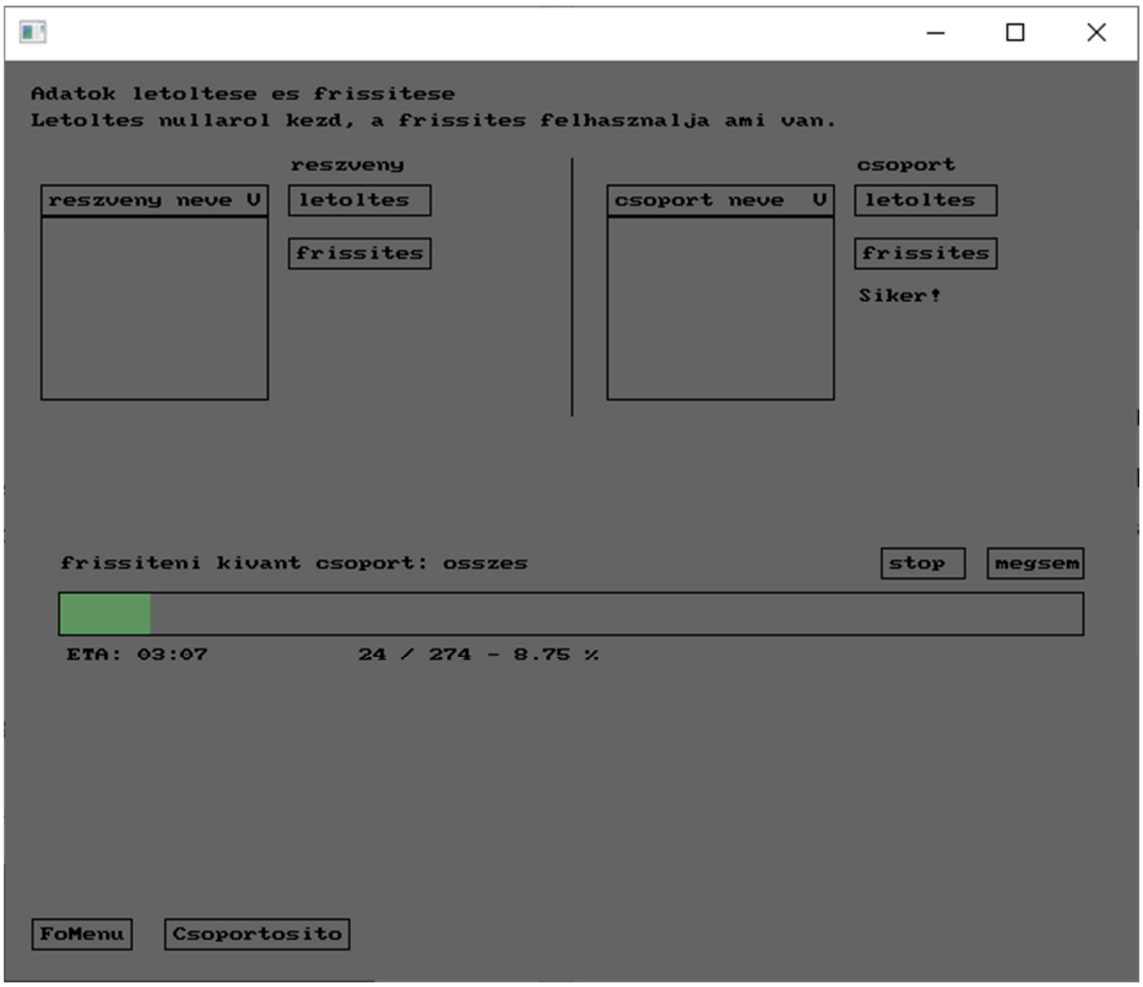
Csak navigálásért felel a menük között és a megengedett szálak számát lehet itt módosítani.



1. ábra Főmenü

3.2.2 Adatok frissítése menü

Itt lehet a részvényeket és csoportok adatait letölteni, meglévőeket frissíteni. Továbbá a folyamat előre haladtát lehet nyomon követni. A letöltési folyamatot elindítani, szüneteltetni, újraindítani és törölni.



2. ábra Adatok frissítése menü

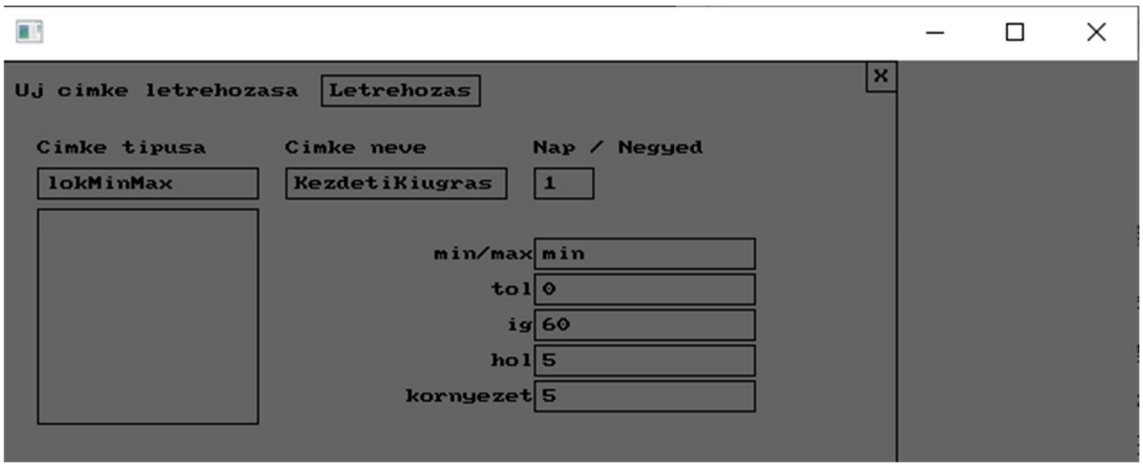
3.2.3 Részvények vizsgálata menü

Ki lehet választani részvényeket, amiket napi vagy negyedéves bontásban lesznek megjelenítve. Lehet lépkedni napok, negyedévek között vagy kiválasztani egy adottat. Lehet százalékos eltérést vizsgálni, akár az előző naphoz, negyedévhez képest. Adott részvény tetszőleges időszakát lehet egy sorban megjeleníteni. Az egér görgőjét lenyomva lehet pásztázni.

3. ábra Részvények menü

3.2.4 Új címke létrehozása - felugró ablak

A részvények menüben lehet ezt az ablakot előhozni. A részvények megtekintése közben a legkézenfekvőbb új címkéket létrehozni. A létrehozott új címkét fájlba kimenti.



4. ábra Új címke menü

4 Hivatkozások

[1] J. Chen, „Investopedia,” 04 04 2023. [Online]. Available: https://www.investopedia.com/terms/t/technical-analysis-of-stocks-and-trends.asp. [Hozzáférés dátuma: 08 12 2023].

[2] „Simple DirectMedia Layer,” június 2023. [Online]. Available: https://www.libsdl.org/.

[3] B. blog, „Memory Mapping,” https://bertvandenbroucke.netlify.app/2019/12/08/memory-mapping-files/, 2019.

[4] S. Heeren, „stackoverflow,” 12 2019. [Online]. Available: https://stackoverflow.com/questions/17465061/how-to-parse-space-separated-floats-in-c-quickly. [Hozzáférés dátuma: 08 12 2023].