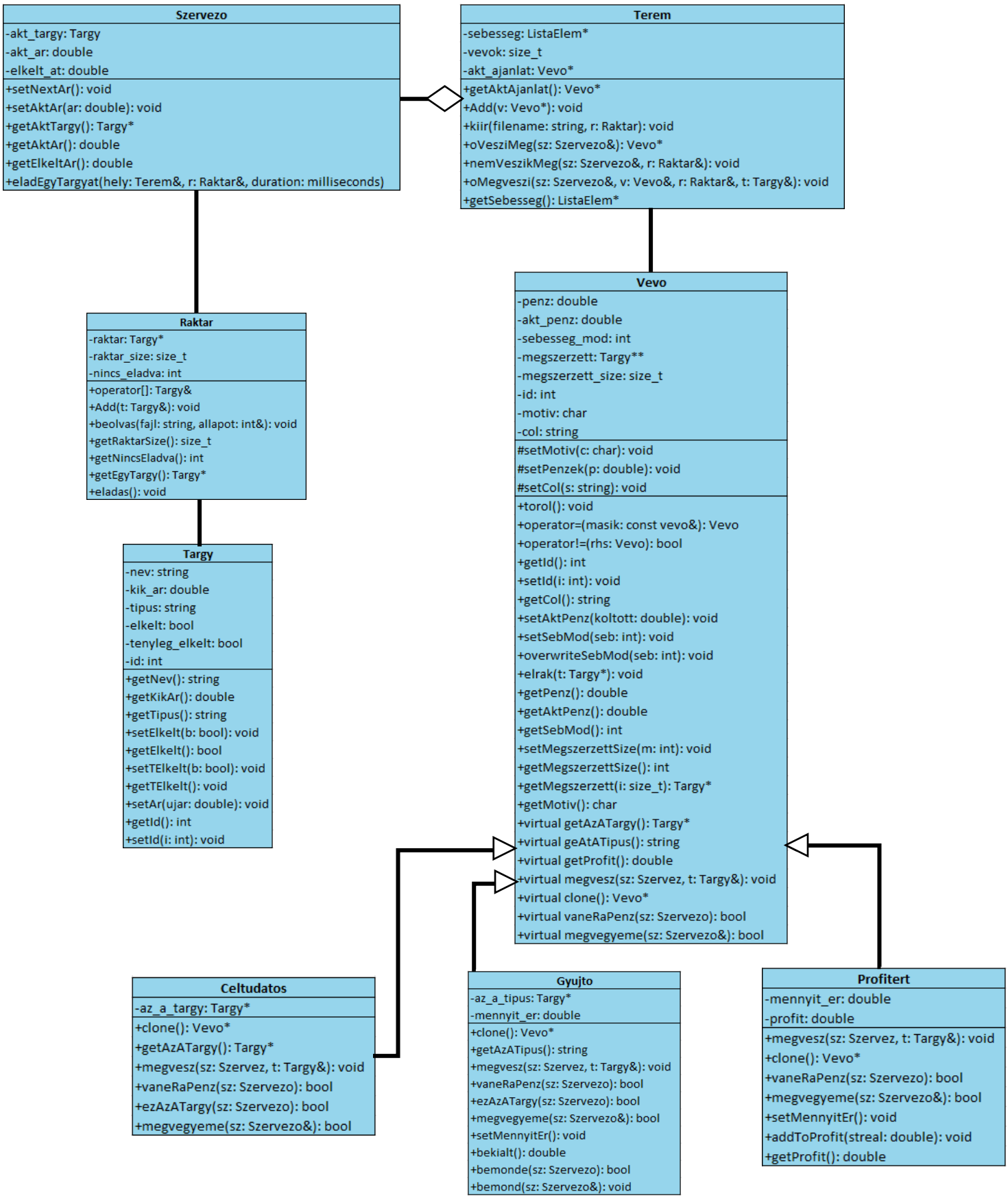
Árverés szimuláció



Tartalom

Szimuláció leírása

Árverés folyik egy teremben. Az árverést egy szervező vezényli, aki a raktárból választja a tárgyakat random sorrendben. Ezekre a tárgyakra a teremben lévő vevők licitálnak. Minden vevőnek megvannak a saját preferenciái. Van, aki egy tárgyra feni a fogát, van, aki valamilyen tárgyak gyűjtője, így csak azokra a tárgyakra fog licitálni és van, aki mindent megvenne egy jó áron.

Tárgy

Leírás

A tárgyakat próbálják megszerezni a vevők. Egy tárgynak van neve, kikiáltási ára, típusa és ID-je. Emellett eltárolják, hogy sorra kerültek-e az eladásban és hogy meg lettek e véve.

Adatok

nev

Ez a tárgy neve.

kik_ar

Ez a tárgy kikiáltási ára.

tipus

Ez a tárgy típusa.

elkelt

Ez tárolja, hogy eladásra került-e ez a tárgy.

tenyleg_elkelt

Ez tárolja, hogy ez a tárgy meg lett e véve.

Függvények

setterek és getterek

Raktár

Leírás

A raktár egy tárolója a tárgyaknak. A szervező ebből választja ki a tárgyakat, amiket elad. Emellett azt is tárolja, hogy hány tárgy nincs még eladva a raktáron belül.

Adatok

raktar

Ez egy tárgy tömb első elemére mutató pointer.

raktar_size

Ez tárolja, hogy hány tárgy van ebben a raktárban.

nincs_eladva

Ez fontos, mert azt tárolja, hogy hány tárgy nincs még eladva. Amint ez az érték 0, az árverés a végére ér.

Függvények

operator[]

Ez visszaadja a raktárban tárolt tárgy tömb index-edik elemét.

Add

Ez a függvény megkap egy tárgy referenciát és ezt a tárgyat belerakja a tárgy tömbbe.

beolvas

Ez a függvény megkapja a beolvasandó fájl nevét és az állapotváltozót. Ha létezik a fájl, akkor a fájlból adatokat olvas be, amiket belerak a raktár tömbjébe. Ha a fájl nem létezik, akkor átállítja az állapot változót.

Szervező

Leírás

A szervező vezeti az egész árverezést. Ő választja ki azt, hogy melyik tárgyat adja el, ő adja meg a következő árat. Tárolja, hogy melyik tárgyat adja el most, az aktuális árat és azt az árat, amit már egy vevő bemondott.

Adatok

akt_targy

Ez az a tárgy, amit legutoljára választott ki, ezt adja el most.

akt_ar

Ez az az ár amire a vevők tudnak reagálni.

elkelt_ar

Ez az az ár, amit már bemondott egy vevő.

Függvények

getNextTargy

Kiválaszt egy tárgyat a raktárból, ami még nem lett eladva és azt beállítja akt_targy-nak. Ennek a tárgynak az alap ára lesz a kikiáltási ár.

eladEgyTargyat
Ez a függvény megmozgat mindent. Paraméternek megkap egy termet, egy raktárt és egy időegységet. A paraméterként kapott terem vevői között eladja az aktuális tárgyat majd frissíti a raktár és a teremben lévő vevők adatait. Az időegység az csak azért szükséges, mert ez a függvény ír a konzolba és ahhoz, hogy egy chat szobában üzenetekhez hasonló időzítéssel tudja kiírni a teremben elhangzókat és ezt a változót a felhasználó adja meg a standard bemeneten.

Terem
Leírás
A terem tárolja a vevők láncolt listáját, és annak a vevőnek a pointerét, aki adta az utolsó ajánlatot. Emellett megjegyzi, hogy hány vevő van a listában.

Adatok
sebesseg
Ez a láncolt lista, amiben a vevők a sebesség modifier-jük alapján vannak sorba rakva a listához való hozzáadásuk pillanatában.

vevok
Ez a változó megjegyzi, hogy milyen hosszú a vevők listája. A haszna az, hogy amikor egy új vevőt rakunk a listába akkor a vevő ID-ja megkapja ezt az értéket. Ez azért kell mert a listában lévő index-e a vevőknek nem tükrözi a létrehozásuk sorrendjét.

akt_ajanlat
Ez az aktuális ajánlatot tevő vevő pointere. Hasznos amikor egy tárgynak a kikiáltási ára olyan magas, hogy egyik vevő sem tartotta az árat vagy ha már senki se fizetné az aktuális tárgyért az aktuális árat.

Függvények
Add
Ez a függvény kap egy vevő pointert és beszúrja a listába a sebesség modifier-je alapján a megfelelő helyre.

kiir
Ez a függvény kiírja az árverés minden adatát a filename által megadott fájlba. A fájlba írja minden vevő pénzét, adatai és az általa megvett tárgyak adatait. Emellett még kiírja azoknak a tárgyaknak az adatait is, amiket senki sem vett meg.

oVesziMeg
Ez a függvény megmondja, hogy melyik vevő az, aki először mondja be az aktuális árat. Ha ez a pointer nullpointert ad vissza akkor az aktuális árt senki se fizetné ki az aktuális tárgyért.

nemVeszikMeg
Ha egy tárgy kikiáltási ára túl magas és ezért egyik vevő se venné meg akkor ez a függvény hívódik. Ez a függvény kezeli azt az esetet. Átállítja az aktuális tárgy adatait, de nem rakja bele egyik vevő megszerzett tömbjébe se.

oMegveszi
Ez a függvény a nemVeszikMeg ellenkezője. Neki paraméterként küldött objektumokban frissíti az adatokat és a paraméterként kapott vevő megszerzett listájára rakja az aktuális tárgyat.

Vevő

Leírás

Ez a fő vevő osztály. Eltárolja egy vevő pénzét az árverés elején és annak folyamán, a vevő gyorsaságát, az általa megszerzett tárgyak tömbjét, ennek a tömbnek a méretét és a vevő ID-ját. Emellett eltárol kiírást segítő paramétereket is.

Adatok

penz

Ez a vevő pénze az árverés elején. Ennek csak a profit motiváltságú és a gyűjtő vevőknél van szerepe.

akt_penz

Ez tárolja a vevő pénzt ebben a pillanatban.

sebesseg_mod

Ez jelöli azt, hogy egy vevő mennyire gyors. Ez alapján vannak rendezve a vevők a sebesség listában

megszerzett

Ez egy tárgy pointer tömb, ami azoknak a tárgyaknak a pointerjeit tárolja el, amiket ez a vevő vett meg.

megszerzett_size

A megszerzett tömb mérete.

id

A sebesség listába való feltöltésükkor kapják meg ezt az értéket. A létrehozásuk sorrendjét jelöli.

motiv

Ez azt teszi lehetővé, hogy a centralizált kiíró függvény tudja, hogy melyik függvényeket érdemes meghívni erre a vevőre. (pl ne kérje le egy céltudatos vevőtől, hogy mennyi profitja van)

col

Ez csak azért van, hogy a konzolban lássuk azt, hogy melyik vevőnek mi a motivációja. Ez színesebbé teszi a megjelenítést.

Függvények

torol

Törli a dinamikusan létrehozott vevőt.

operator!=

Ha ennek a vevőnek a memóriacíme ugyan az, mint az operátor jobb oldalán lévő vevőnek akkor hamis, egyéb esetben igaz.

setAktPenz

Ez a függvény levonja a vevő aktuális pénzéből a paraméterként kapott összeget.

setSebMod

A paraméternek megadott számmal növeli a sebesség modifier-t.

overwriteSebMod

Felülírja a sebesség modifier értékét a paraméterként kapott értékre.

elrak

A paraméterként kapott tárgy pointerét belerakja a vevő megszerzett tömbjébe.

Virtuális függvények

- getAzATargy
- getAzATipus
- getProfit
- megvesz
- clone
- vaneRaPenz
- megvegyeme

Céltudatos

Leírás

Ezek azok a vevők, akiket az motivál, hogy megszerezzék azt az egy tárgyat amiért az árverésre jöttek.

Adatok

az_a_targy

Ez annak a tárgynak a pointere, amit meg akarnak szerezni.

Függvények

clone

Ez a függvény visszaadja a vevőnek egy dinamikusan foglalt változatát.

megvesz

Ebben az esetben a függvény elrak-ot hív és levonja a megszerzett tárgy árát a vevő aktuális pénzéből.

vaneRaPenz

Ebben az esetben a függvény megnézi, hogy van-e a vevőnek annyi pénze, mint amennyibe az aktuális tárgy most kerül.

ezAzATargy

Ez a virtuális függvény ennél a vevő típusnál hasznos. Hogyha az aktuális tárgy az, amit ez a céltudatos vevő meg akar venni akkor igaz, egyébként hamis.

megvegyeme

Ebben az esetben a függvény igaz, ha ez az a tárgy, amit meg akar venni és van is rá pénze, egyéb esetben hamis.

Gyűjtő

Leírás

Ezek azok a vevők, akiket az motivál, hogy megszerezzék az összes olyan típusú tárgyat, amiket gyűjtenek. Emellett minden tárgyról eldöntik, hogy nekik mennyit érne, ha olyan lenne a típusuk, amit gyűjtenek.

Adatok

az_a_tipus

Ez az a típus, amit ez a gyűjtő gyűjt.

mennyit_er

Ez tárolja, hogy pénzének hány százalékát éri meg a vevőnek ez a tárgy.

Függvények

clone
Ez a függvény visszaadja a vevőnek egy dinamikusan foglalt változatát.

megvesz
Ebben az esetben a függvény elrak-ot hív és levonja a megszerzett tárgy árát a vevő aktuális pénzéből.

vaneRaPenz
Ebben az esetben a függvény megnézi, hogy van-e a vevőnek annyi pénze, mint amennyibe az aktuális tárgy most kerül. És azt is, hogy a jelenlegi ár kisebb-e, mint amennyiért neki megéri. Ha mind a két feltétel igaz akkor igaz, egyéb esetben hamis.

ezAzATius
Ez a virtuális függvény ennél a vevő típusnál hasznos. Hogyha az aktuális tárgy típusa az, amit ez a gyűjtő vevő gyűjt, akkor igaz, egyébként hamis.

megvegyeme
Ebben az esetben a függvény igaz, ha ez az a tárgy megfelelő típusú, van is rá pénze és meg is éri neki, egyéb esetben hamis. Itt egyéb funkciója is van. Ez a függvény lehetőséget ad arra, hogy a gyűjtő bemondjon egy új árat.

setMennyitEr
Újra sorsolja a mennyit_er értékét.

bekialt
Visszaadja a két harmadát annak az összegnek, amennyit szerinte ez a tárgy ér

bemonde
Igaz, ha a bekialt összeg nagyobb, mint a jelenlegi ár, egyéb esetben hamis.

bemond
Ez a függvény akkor van meghívva, amikor e a gyűjtő kevésnek találja a mostani árat. Ez a függvény beállítja az aktuális árat arra, amit ez a gyűjtő gondol jónak, ezzel felgyorsítva az árverezést.

Profitért

Leírás
Ezek azok a vevők, akiket az motivál, hogy minél több profitjuk legyen az árverezés végén. Minden tárgyról eldöntik, hogy nekik mennyit érne és afölé az ár fölé ők már nem mennek.

Adatok

mennyit_er
Ez tárolja, hogy pénzének hány százalékát éri meg a vevőnek ez a tárgy.

profit
Ez tartja számon, hogy mennyi profitot szerzett ez a profit motivált vevő.

Függvények

megvesz
Ebben az esetben a függvény elrak-ot hív, levonja a megszerzett tárgy árát a vevő aktuális pénzéből és a szerzett tárgy árának és az általa megért árnak a különbségét hozzáadja a profit-hoz

clone
Ez a függvény visszaadja a vevőnek egy dinamikusan foglalt változatát.

vaneRaPenz
Ebben az esetben a függvény megnézi, hogy van-e a vevőnek annyi pénze, mint amennyibe az aktuális tárgy most kerül. És azt is, hogy a jelenlegi ár kisebb-e, mint amennyiért neki megéri. Ha mind a két feltétel igaz akkor igaz, egyéb esetben hamis.

megvegyeme
Ebben az esetben a függvény értéke ugyan az, mint a vaneRaPenz-nek.

addToProfit
Ez a paraméternek megadott értékkel megnöveli a profit értékét.

Tesztelés
1.Teszt
Ez leteszteli, hogy az első tárgy jól lett-e létrehozva.

2.Teszt
Ez leteszteli, hogy az második tárgy jól lett-e létrehozva.

3.Teszt
Ez leteszteli, hogy az harmadik tárgy jól lett-e létrehozva.

4.Teszt
Ez leteszteli, hogy a vevők pénze jó értékekre lettek beállítva.

5.Teszt
Ez leteszteli, hogy a vevők motiv-ja jó értékekre lettek beállítva.

6.Teszt
Ez leteszteli, hogy a vevők sebesség modifier-je jó értékekre lettek beállítva.

7.Teszt
Ez leteszteli a tesztben lévő vevők egyedi paramétereit.

8.Teszt
Ez magába foglalja az első eladást. Ez leteszteli a működését számos függvénynek, például az eladEgyTargyat és a oVesziMeg függvényeket. Emellett azt is megnézi, hogy bekerült-e a megfelelő vevő megszerzett tömbjébe a megfelelő tárgy pointere.

9.Teszt
Ez a második eladást teszteli hasonló módon, mint a 8. teszt az első eladást.

10.Teszt
Ez a harmadik eladást teszteli hasonló módon, mint a 8. teszt az első eladást.

11.Teszt
Ez leteszteli, hogy egy vevő annyi pénzt költött-e, mint amennyibe az általa megvett tárgyak kerültek.

Main

A kód elsőként bekéri a futtatás célját vagy a vevők eloszlását. Ez a menet változó értéke

0: teszteset

1: a vevők aránya a teremben 1:1:1 (céltudatos:profitért:gyűjtő)

2: a vevők aránya a teremben 3:2:1 (céltudatos:profitért:gyűjtő)

3: a vevők aránya a teremben 3:1:2 (céltudatos:profitért:gyűjtő)

A program létrehoz egy új termet, szervezőt, raktárt, időegységet és egy stringet.

Hogyha a bemenet nem 0, akkor a program bekéri:

- az időegységet, amivel a kiírásokat időzíti (ms)
- a beolvasni kívánt fájl nevét, amiben a raktárba olvasandó tárgyak adatai vannak tabulátorral elválasztva (név\t ár\t típus)
- a fájl neve, amibe a program az árverés szimulálása után kiírja az adatokat (vevők adatai, mit nem vett meg senki)

A menet értékétől függően létrehoz 30 vevőt és berakja őket a terem listájába.

Ez után, ha a menet > 0 akkor elad egy tárgyat addig amíg van olyan tárgy a raktárban, amit még nem adott el a szervező. A ciklus után pedig kiírja az adatokat a megadott nevű fájlba.