Tartalomjegyzék

Nagyházi specifikáció

1. Feladat

Készítsen objektummodellt nagy tömegű testek (bolygók) kölcsönhatásának modellezésére!
Legyen lehetőség minden test tömegének és kezdőpozíciójának egyszerű megadására. Ezt követően az egymással kommunikáló objektumok határozzák meg rájuk ható erőket és helyváltoztatással határozzák meg a nyugalmi pozíciójukat!
Javaslat: Az objektumok kommunikációját szervezzük gyűrűbe. A gyűrűben egy N elemű vektort körbeküldve mindenki megismerheti a szomszédok helyzetét és töltését, így miden objektum ki tudja számolni a rá ható erőket, amiből számítható az elmozdulás. Az iteráció addig folytatódik, míg be nem áll az egyensúlyi helyzet, vagy az iterációs ciklusszám el nem ér egy előre megadott értéket.
Olyan modellt tervezzen, hogy tetszőleges számú test is modellezhető legyen !

2. Pontosított specifikáció

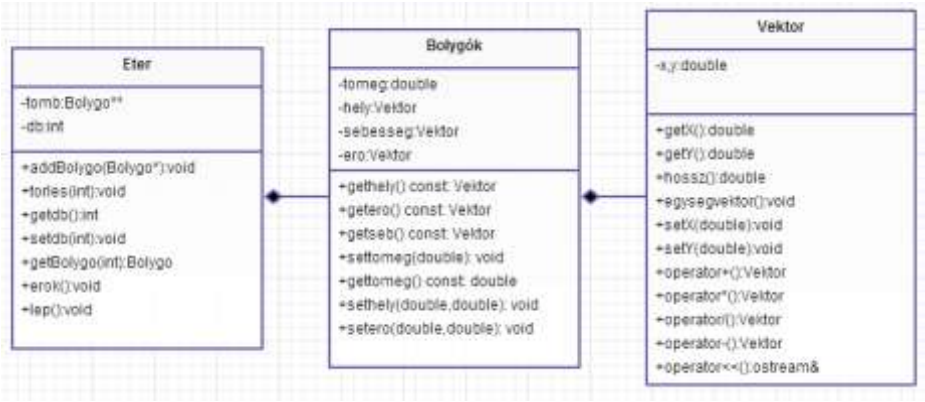
Egy objektummodellt kell készítenem, ami nagy tömegű testek kölcsönhatásait tudja modellezni. Nem volt pontosan meghatározva ezért 2D-ban fogom ezek a testeket kezelni.

Könnyedén létre tud hozni testeket (bolygókat) kezdeti pozícióval és tömeggel. Minden testet az egyszerűség kedvéért pontnak tekintek. Minden test meg tudja határozni a rá ható gravitációs erőket. Így mindegyik test ki tudja számolni azt hogy hol lesz a következő helyzete. Tehát minden test tudja a másik test pozícióját és tömegét. Véges sok testet kell tudnom tárolni. Ha egy test elindul akkor lesz egy lendülete és ezt fogja változtatni a rá eső erők eredője.

3.Terv

Osztályok:

Vektor osztály:
-x,y:double
fuggvenyek
+getX():double
+getY():double
+hossz():double
+egysegvektor():void



+setX(double):void
+setY(double):void
+operator+():Vektor
+operator*():Vektor
+operator/():Vektor
+operator-():Vektor
+operator<<():ostream&

Bolygó osztály:
-tomeg:double
-hely:Vektor
-sebesseg:Vektor
-ero:Vektor
függvények
+gethely() const: Vektor
+getero() const: Vektor
+getseb() const: Vektor
+settomeg(double): void
+gettomeg() const: double
+sethely(double,double): void
+setero(double,double): void

Eter osztály:
-tomb:Bolygo**
-db:int
+addBolygo(Bolygo*):void
+torles(int):void
+getdb():int
+setdb(int):void
+getBolygo(int):Bolygo
+erok():void
+lep():void

Főbb algoritmusok:

Eter::Erok()

```
for(int i=0;i<db;i++)
   {
      Vektor irany;
      for(int j=i+1;j<db;j++)
      {
         if(j!=i)
         {
         irany = tomb[j]->hely - tomb[i]->hely;                    //Az erőnek ebbe az irányba kell néznie
         irany.egysegvektor();                                     //Egységvektort gyártok belőle, hogy meg legyen az erő iránya
                                                                   //Az irányt megszorzoma mérettel
         irany = irany * ( tomb[i]->gettomeg() * tomb[j]->gettomeg()/(pow((tomb[i]->hely-tomb[j]->hely).hossz(),2)));
         tomb[i]->ero = tomb[i]->ero +irany;                       //Erő hozzáadása  a többi erőhöz
         tomb[j]->ero = tomb[j]->ero - irany;                      //Ellenerő hozzáadása
```

```
        }
    }
}
for(int i = 0; i< db;i++)
    tomb[i]->ero = tomb[i]->ero*G;                    //Itt szorzom meg a G-vel mert így kevesebbet kell számolnia.
```

Eter::lep()

```
for(int i = 0; i< db; i++)
{
    tomb[i]->seb = ((tomb[i]->ero/tomb[i]->gettomeg()) + tomb[i]->seb) ;   //sebesség az erő hatására megváltozik
    tomb[i]->hely = (tomb[i]->hely + tomb[i]->seb );                       //a hely a sebesség hatására megváltozik
}
```

4.A test program

A tesztprogramba 4 féle tesztet tettem:

- Az első a naprendszer bolygói egy napközéppontú koordináta rendszerben. Nincs benne az összes bolygó csak a Naptól a Föld-ig. Az eredeti adataikkal vannak benne a tesztprogramban SI mértékegységben megadva. 180 napnyi keringést mutat be a program, minden másodpercben kiszámolja de csak a napok elejét írja ki.
- A második teszt eset két darab bolygót tartalmaz amik összeütköznek és egybeolvadnak a tömegük összeadódik a sebességük pedig a lendületül alapján változik.
- A harmadik teszt esetben nem raktunk bolygókat az éterbe, mégis szeretnénk kiírni ilyenkor kiírja a program hogy nincs bolygó az éterben.
- A negyedik esetben hibásan indexeljük meg az étert. Ezért out_of_range hibát dob, amit elkap és kiírja hogy indexelési hiba.

5.Tesztelési dokumentáció

A naprendszert és a bolygók összeütközését grafikus környezetben is teszteltem, de az első tesztben is látszik, legjobban a Merkúron mivel az kering a leggyorsabban, hogy az x koordinátája pozitívból negatívba megy körül belül ugyan akkora mértékben majd vissza jön pozitívba és ez ismétlődik. Szintén az y koordinátájával.

Két bolygó ütközése a második teszt eset. Lehet látni hogy amikor elindul a program akkor még csak egész lassan kezdenek egymás felé közelíteni. Aztán pedig egyre gyorsabban, ez egyre növekvő gyorsulás mivel minél közelebb vannak egymáshoz annál jobban vonzzák egymást. Végül amikor egymáshoz közel kerülnek egy bolygóvá olvadnak és mivel az 1 bolygó stabil helyzet így a program leáll. A teszt esetben ez a bolygó: x= 8.61492*10^8 y= 0 koordinátáknál olvadtak egybe.

A harmadik és negyedik tesztesetben az elvártaknak megfelelően a program kiírja az adott üzeneteket. A negyedik tesztesetben ezek szerint a hiba elkapása sikeres volt.