OEP beadandó dokumentáció

Név: Kmeczó András

Neptun kód: ROXFTL

**Feladat:**

1. Feladat: Ismerjük a **légkör** egymás felett elhelyezkedő ózon, oxigén, és széndioxid anyagú légrétegeit, amelyek vastagsága a légköri viszonyoktól (zivataros, napos, egyéb) függően változik. Amikor egy légköri réteg anyagának egy része átalakul, akkor ez az anyagmennyiség fölszáll, és vastagítja a felette lévő első ugyanolyan anyagú, de nem felszálló réteget. Ha nincs fölötte ilyen réteg, akkor a légkör legtetején új réteget képez, ha legalább fél kilométer vastag. Egy rétegnek sem csökkenhet a vastagsága fél kilométer alá. Ha ez mégis megtörténne, akkor ez a réteg is felszáll, és egyesül a fölötte lévő első ugyanilyen anyagú réteggel. Ha nincs ilyen, akkor megszűnik.

A folyamat során először egymástól függetlenül reagálnak az egyes légrétegek az aktuális időjárási viszonyra, utána rétegenként alulról felfelé haladva felszállnak az újonnan keletkeztek anyagmennyiségek, illetve a túl vékony rétegek.

Az egyes anyagok az alábbi módon reagálnak a különböző időjárási viszonyokra.

|  |  |  |  |
| --- | --- | --- | --- |
| **anyag** | ***ózon*** | ***oxigén*** | **széndioxid** |
| ***zivataros*** | - | 50% ózonná | - |
| ***napos*** | - | 5% ózonná | 5% oxigénné |
| ***más*** | 5% oxigénné | 15% széndioxiddá | - |

Készítsen használati eset diagramot, ahol az időjárás és a légkör szempontjából lényeges eseteket, valamint ezek kapcsolatát ábrázolja. Adjon meg olyan szekvencia diagramot, amely az időjárás, a légkör, és a légkör rétegei közötti kommunikációban érintett metódusokat mutatja meg. Rajzolja fel a légkör időjárásának és egy terület felszínének állapotgép diagramját! Készítse el az osztály diagramot! Használjon állapot és látogató tervezési mintákat.

Implementálja a modellt, és oldja meg az alábbi feladatot: ***Addig szimuláljuk a folyamatot, amíg el nem fogy valamelyik anyag teljesen a légkörből. Körönként mutassuk meg a légrétegek összes tulajdonságát!***

A program egy szövegfájlból olvassa be a légkör adatait! Az első sorban a változó légköri viszonyok találhatók egy karaktersorozatban (z - zivatar, n - napos, m – más). Ha a szimuláció a karaktersorozat végére ér, az elejéről folytatja. A következő sorok tartalmazzák alulról felfelé haladva a légrétegek adatait szóközökkel elválasztva: anyaga (ezt egy karakter azonosítja: z - ózon, x - oxigén, s - széndioxid), és vastagsága.

A program kérje be a fájl nevét, majd jelenítse is meg a tartalmát. (Feltehetjük, hogy a fájl formátuma helyes.) Egy lehetséges bemenet:

mmmmnnznnmm

z 5

x 0.8

s 3

x 4

Készítsen teszteseteket, és hozzon létre ezek kipróbálására automatikusan tesztkörnyezetet!

**Használati eset diagram:**

A program fő mozgatórugója az időjárás változása, és a hozzá kötődő változások a legköri rétegekben. Az időjárás változásakor minden réteg sorban változik, majd létrejöhet belőle egy új réteg, ami felszáll, vagy törlődik ha nincs megfelelő felsőbb réteg. Az eredeti réteg is felszáll, ha túl vékony, valamint törlődik, ha nincs felette megfelelő réteg.



**Szekvencia diagram:**

A program egy végtelen ciklusban fut, mely a CheckOver() teljesülése esetén lép csak ki a ciklusból. A ciklus első lépése, hogy az aktuális IWeatherCondition objektumot felhasználva végigmegyünk minden AtmosphereLayeren, majd a HandleLayer() segítségével minden légréteget frissítünk az időjárás hatásaival. A lista végére érve meghívjuk a HandleCreatedLayers() függvényt, mely a létrejött és esetleg túl vékonyra szűkített légrétegek felszállását illetve esetleges eltűnését kezeli. A PrintState() ezután kiírja az aktuális kör állapotát, majd a NextCondition() a következő IWeatherConditiont kéri le és készíti elő a következő ciklus futásához.



**Állapotgép diagram**

A programhoz tartozó állapotgép (StateMachine) tárolja a légrétegeket, valamint az időjárási viszonyokat, amiken végigmegy a főprogram.



**UML diagram:**



**Megoldási terv:**

* Az argumentumban megadott fájlt betöltjük, majd a Main() függvényben feldolgozzuk az adatokat
* A kapott listákkal meghívjuk a StateMachine.Init() függvényt, amivel inicializáljuk a programot, majd a Start() függvénnyel elindítjuk a szimulációt
* Ha elfogyott teljesen valamelyik légrétegfajta, akkor befejezzük a program futását

**Tesztelési terv:**

* A tesztelésnél arra kell figyelni, hogy a program leállásakor számolt légrétegek megfeleljenek az elvárt kimenetnek, illetve a köztes lépéseket is lehet tesztelni, hogy még pontosabb képet kapjunk a program helyességéről
* Fontos tesztelni üres, kevés, illetve sok bemenettel, valamint különböző kombinációkkal, hogy lefedjünk minden lehetséges tesztesetet

**Tesztelés:**

* A tesztfájlokat a tests/ mappa tartalmazza, ahol 3 teszteset található, valamint az elvárt kimeneteik. A tesztelésnél a program Teszt módját használjuk, és a Standard Kimenetet átírányítjuk egy StringWriterbe, ahonnan aztán ellenőrizzük, hogy a kimenetek egyeznek-e az elvárttal