HÁZI FELADAT

Programozás alapjai 2.

Feladatválasztás/feladatspecifikáció

Király Bálint
EQF1M0

2023. április 18.

TARTALOM

1. Feladat

Autópálya forgalma

Készítsen objektummodellt az autópálya forgalmának modellezésére! Egy L cellára osztott autópályán N autó van. Egy cellában csak egy autó tartózkodhat egyszerre, így L-N cella üres. Minden autónak van egy egész értékű sebessége. A szimulációt ciklusonként végezzük. Minden ciklusban minden autóra elvégezzük a következő műveleteket:

1. Ha egy autó sebessége még nem érte el a maximumot (5), akkor a sebességét eggyel megnöveljük.
2. Ha egy autó előtt levő üres cellák száma (az előtte levő autóig) kisebb, mint a sebessége, akkor az autó sebességet lecsökkentjük az előtte levő üres cellák számának megfelelő értékre.
3. Egy adott p(=0.15) valószínűséggel csökkentjük a mozgó autók sebességet eggyel. (Vezetők figyelmetlensége).
4. Minden autót előremozgatunk annyi cellával, amennyi a sebessége.

Egyszerű karakteres kimenetet feltételezve "rajzolja ki" az autópálya állapotát egy-egy szimulációs ciklus után.
Demonstrálja a működést külön modulként fordított tesztprogrammal! A megoldáshoz **ne** használjon STL tárolót!

2. Feladatspecifikáció

Classes: Simulation (int cycle, size_t highwayCount, Highway* highways[]),
Highway (size_t length, size_t carCount, Car* cars[]),
Car (Speed speed, int pos)

A Szimuláció elején generálódik egy Simulation object, ami még nem tartalmaz autópályát, a ciklus számlálója 0.

A konfiguráló függvényét meghívva egy config fileból beolvassa az autópálya hosszát, az autók számát, illetve helyzetét, és dinamikusan eltárolja létrehozott Highway, és Car objectekben. Az újonnan létrehozott Highway object pointerét hozzáfűzi az autópálya pointereket tároló dinamikus tömbjéhez.

Igény szerint további config fileokból is olvashatunk be adatokat. Minden file egy autópálya kiindulási állapotát tartalmazza.

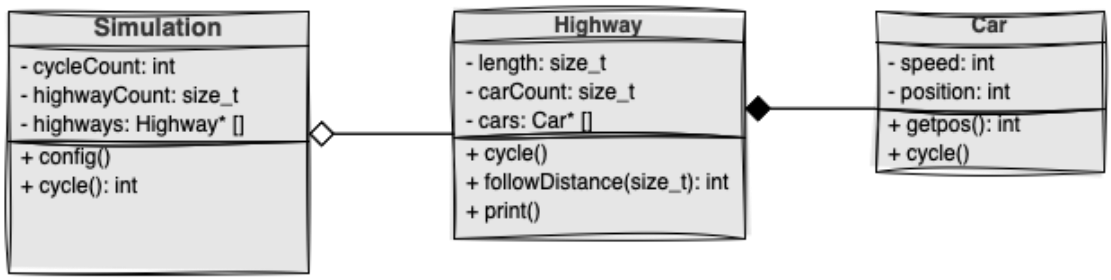
A szimuláció next() tagfüggvényét hívva a ciklus léptethető. Minden autónak kiszámolja a sebességét, és az új helyzetét. A ciklus számláló eggyel nő.

A megjelenítésért a display() tagfüggvény a felelős, mely a szimuláció jelen állását írja ki kimenetre karakteres megjelenítéssel.

A felhasználói felület egészen egyszerű: lehetőség van új autópálya hozzáadására, ciklus léptetésre, illetve a szimuláció befejezésére.

3. Terv

3.1. Osztálydiagram



3.2. Algoritmusok

3.2.1. Konfiguráció

A szimulációhoz hozzáadhatunk uj autópályát külső file-ból. Ehhez a Simulation config tagfüggvényét kell a config file nevével meghívni.

void config(const char* filename)

Config file formátuma:

[L] [N]
[CAR1.POSITION] [CAR1.SPEED]
[CAR2.POSITION] [CAR2.SPEED]
...
...
[CARN.POSITION] [CARN.SPEED]

A konfiguráció algoritmusa:

OPEN file

Highway FOGLALÁS, ahol length = L, carCount = N

highwayCount ++

cars tömb FOGLALÁS

MINDEN Car-ra:

position = car[i].position, speed = car[i].speed

CIKLUS VÉGE

CLOSE file

3.2.2. Ciklus léptetés

A cycle számlálót a Simulation class tartalmazza, így ennek a tagfüggvényét hívjuk meg minden ciklusléptetés alkalmával. Meghívja a többi osztály saját cycle függvényét is, és visszaadja a ciklusszámot.

int Simulation::cycle()

MINDEN Highway-re:

MINDEN Car-ra:

HA speed < 5: speed ++

HA speed > followDistance: v = followDistance

HA humanFactor: speed —

position = position+speed

CIKLUS VÉGE

print

CIKLUS VÉGE

RETURN cycleCount

Az aláhúzott kifejezések más függvényekbe vannak kiszervezve, ezeket is meg kell valósítani.

3.2.3. Emberi tényező

Globális függvény. p valószínűséggel ad igazat.

bool humanFactor()

r = RANDOM szám 0-99 közt

HA r < p*100: RETURN true

ELSE: RETURN false

3.2.4. Követési távolság

Megnézi a következő indexű autó pozícióját, és kivonja belőle az aktuális indexű autó pozícióját.

int Highway::followDistance(size_t index)

RETURN cars[i+1]->getpos() - cars[i]->getpos()

3.2.5. Megjelenítés

Minden ciklusban a kimeneti streamre kirajzolja a szimuláció aktuális állapotát.

void Highway::print(std::ostream& os)

ixcar = 0

CIKLUS pos = 0-length közt:

HA cars[ixcar]->getpos() == pos:

PRINT autó

ixcar ++

HA nem: PRINT blank

CIKLUS VÉGE

3.2.6. Main

Az egyes funkciókat bizonyos karakterekkel lehet vezérelni. Az autópálya hozzáadása [1], a ciklus léptetés [space], a program bezárása [2] lehetőségek érhetőek el. A program magját a következő algoritmus alkotja:

Simulation

PRINT vezérlés[1,2]

AMÍG nincs EOF, és olvasunk az input stream-ről:

HA [1]: Simulation::config()

HA [space]: Simulation::cycle()

HA [2]: EXIT

PRINT vezérlés

CIKLUS VÉGE