Grunnatriði stýrikerfa
Forritunarverkefni 1 - fyrri hluti, skipulag

February 2017

English follows.

Afurðin

Afurðin í þessum hluta er skjal sem nefnir 1) þær semafórur sem búnar verða til í kerfinu, 2) þá atburði (t.d. lyfta kemur á hæð, lyfta yfirgefur hæð, persóna kemur á hæð) sem gerast í kerfinu og 3) hvaða aðgerðir er kallað á á semafórunum við hvern atburð. Skýringarmyndir og kerfislýsingar eru einnig velkomnar.

Að sækja og skila verkefni

Verkefnið gildir ekki til einkunnar, en nemendur fá endurgjöf frá kennara á þær hugmyndir sem lýst er í skjalinu. Nemendur skila PDF skjali inn á myschool. Engu máli skiptir hvort allir nemendur í hóp skila eða hvort hópar sem skila eru þeir sömu og skila á endanum forritunarverkefninu. Ekki skila mörgum eintökum af sömu skjölunum.

Vandamálið

Fólk kemur inn í hús og vill taka lyftu milli hæða. Hver lyfta er útfærð sem þráður og hver persóna er útfærð sem þráður. Á hverjum tímapunkti inniheldur kerfið því þráð fyrir hverja lyftu og þráð fyrir hverja persónu sem hefur komið inn í kerfið og hefur ekki náð að ljúka keyrslu með því að komast inn á hæðina sem hún var á leiðinni á.

Þræðirnir eiga að sjá um að bíða á réttum stöðum eftir að þeir komist áfram í kerfinu og að losa um þá lása sem eðlilegt er að losa á hverjum tímapunkti til þess að aðrir þræðir komist einnig áfram í kerfinu.

Í þessari hönnunarvinnu má gera ráð fyrir að lyftuþræðirnir séu tilbúnir og byrjaðir að keyra og að utanaðkomandi áhrif setji í gang persónuþræðina og

velji á hvaða hæð þeir byrja og á hvaða hæð þeir ætla. Þið þurfið þó að hanna alla virkni þráðanna í keyrslu.

Mismunandi flækjustig kerfisins eru eftirfarandi:

- Ein lyfta, tvær hæðir, allir koma inn á neðri hæð og yfirgefa lyftuna á efri hæð.
- Ein lyfta, óákveðinn fjöldi hæða, allir koma inn á neðri hæð og yfirgefa lyftuna á einhverri hæð. Persónuþráðurinn veit frá upphafi á hvaða hæð hann vill yfirgefa lyftuna.
- Ein lyfta, tvær hæðir, persónur geta komið inn á hvorri hæðinni sem er og yfirgefa hana á hinni hæðinni.
- Ein lyfta, óákveðinn fjöldi hæða, persónur geta komið inn á hvaða hæð sem er og yfirgefið lyftun á hvaða hæð sem er. Persónuþráður veit frá upphafi á hvaða hæð hann vill yfirgefa lyftuna.
- Óákveðinn fjöldi lyfta, óaákveðinn fjöldi hæða, persónur koma inn og yfirgefa lyftuna á hvaða hæð sem er. Persóna kemur inn á hæð og getur notað hvaða lyftu sem er til að ferðast.

Hvert kerfi má útfæra þannig að pláss sé fyrir eina manneskju í lyftunni eða þannig að pláss sé fyrir 6 manneskjur í lyftunni. Þið megið velja hvenær þið bætið þessu flækjustigi inn, en á endanum eiga öll stigin að gera ráð fyrir 6 manneskjum í lyftu í einu.

Endilega bætið við millistigum flækju ef það hjálpar ykkur að skilja vandamálin. Einnig má sleppa stigum í lokaskjalinu ef ykkur þykja þau vera innifalin í öðrum stigum, en bætið samt ekki inn of miklu í einu í hönnunarvinnunni.

Vinnufyrirkomulag

Gott er að byrja á því að teikna mynd af vandamálinu sem þið viljið leysa. Fyrst myndi það vera mynd af einni lyftu sem getur verið á tveimur hæðum, ásamt röð af fólki á neðri hæðinni og útgangi fyrir fólk á efri hæðinni. Fólkið kemur einungis inn á neðri hæðinni og fer einungis út á efri hæðinni.

Næst er gott að skrifa lista yfir þá atburði sem geta gerst í kerfinu, t.d. lyfta yfirgefur efri hæð, lyfta kemur inn á neðri hæð, persóna hefur keyrslu, persóna kemst inn í lyftu, o.s.frv.

Þegar ykkur finnst að listinn yfir atburði sé tilbúinn þá má byrja að telja upp, fyrir hvern atburð, eftir hverju viðkomandi þráður (persóna eða lyfta) þarf að bíða við þessar aðstæður, og hvaða biðlása þarf að losa þegar þessi atburður gerist.

Þetta gæti litið e-n vegin svona út:

- Persóna kemur inn í kerfið
  - Persónuþráður þarf að bíða eftir að lyfta komi á hæðina
  - ...
- Lyfta kemur á neðri hæð
  - Lyftuþráður þarf að losa um biðlása þeirra sem bíða á hæðinni
  - ...
- Lyfta yfirgefur efri hæð
- ...
- ...

Fyrir hverja svona línu þar sem þarf að bíða eftir einhverju eða losa um bið á einhverju þurfið þið að taka eftirfarandi ákvarðanir:

- Er hægt að framkvæma þessa bið eða losun með einfaldri skipun á semafóru?
- ... eða þarf að nota teljara eða aðgang að sameiginlegu minnissvæði?
- Er hægt að nota semafóru sem er þegar í kerfinu?
- ... eða þarf að bæta við nýrri? Teiknið hana þá inn á myndina ykkar.

Og eflaust þarf stundum að taka fleiri ákvarðanir en þetta.

Farið að lokum í gegnum þetta kerfi, myndina ykkar og lista yfir atburði, og gangið úr skugga um að allar biðir og lásar gangi upp. Passið að enginn þráður verði sveltur, að engir lásar valdi endalausum læsingum (deadlock) en jafnframt að enginn þráður komist í gegnum kerfið án þess að bíða eftir réttum tímasetningum.

Þegar eitt svona kerfi er komið, bætið þá við þeim atburðum sem þarf til að kerfið uppfylli næsta flækjustig verkefnisins og farið í gegnum sömu skrefin fyrir þetta kerfi.

Góða skemmtun!

The Product

In this part the product is a document naming 1) the semaphores that will be made and used in the system, 2) the events (elevator stops on floor, elevator leaves floor, person enters floor, etc.) that happen in the system and 3) what operations are called on the semaphores when these events occur. Any images and system descriptions are also welcome.

Getting and returning the assignment

The project will not be graded but students will get feedback from the techer on the ideas described in the document. A PDF document is returned in the project page on myschool. It makes no difference whether all students in a group return or if the groups that return here are the same ones as eventually return the programming assignment. Just do not return many copies of the same document.

The problem

People enter a building and wish to use an elevator to travel between floors. Each elevator is implemented as a thread and each person is implemented as a thread. At any given time the system will include a separate thread for each elevator and each person that has entered the system but not yet finished its run by exiting at the floor it wishes to travel to.

The thread should wait at the appropriate places and unlock certain lock at the appropriate times, to make sure that other threads can also conclude their run correctly.

In the design work you can assume that the elevator threads are already running and that an outside source starts the person threads and lets them know where they are and where they wish to go. You still need to design the running functionality of the threads.

Following are the different levels of complexity:

- Single elevator, two floors, everyone enters at the bottom floor and exits at the top floor.
- Single elevator, any number of floors, everyone enters at the bottom floor and exit at any other floor. The person-thread knows from the beginning at shich floor it wishes to exit.
- Single elevator, two floors, persons can enter at either floor and exit at the other floor.
- Single elevator, any number of floors, persons can enter and exit at any floor. The person-thread knows from the beginning at shich floor it wishes to exit.
- Any number of elevators, any number of floors, persons enter and exit at any floor. Persons enter at a floor and can use any elevator to travel.

Each system can be described either for a single person in an elevator at a time, or for 6 people at a time in an elevator. You can decide when this level of complexity is added, but in the end each system should allow for 6 people in an elevator at a time.

Feel free to add any intermediate levels of complexity if this helps you get your head around the problem. You can also skip levels in the final document if you feel they are included in other levels, but during the design process take care not to add too much at a time.

Process

It is good to start by drawing a simple diagram of the problem you want to solve. This would first be an image of a single elevator that can be on one of two floors, along with a line of people entering on the bottom floor and an exit

for people on the top floor. People only enter on the bottom floor and exit on the top floor.

Next you can write a list of events that can happen in the system, e.g. elevator leaves top floor, elevator stops at bottom floor, person is created, person is allowed into elevator, etc.

When you feel this list is ready it is time to list, for each event, what the thread in question (person or elevator) need to wait for at this point, and what locks need to be opened when this event happens.

The list might look like this:

- Person enters the system
  - Person-thread needs to wait for an elevator to stop at the floor
  - ...
- Elevator stops at bottom floor
  - Elevator-thread needs to unlock threads waiting at the floor
  - ...
- Elevator leaves top floor
- ...
- ...

For each such line where a thread needs to wait or unlock other threads you must make the following decisions:

- Can this wait or unlock be achieved with a simple operation on a semaphore?
- ... or does it need a counter or access to a shared memory object?
- Can you use a semaphore already in the system?
- ... or do you need to add a semaphore? Add it to your diagram.

And without doubt you will sometimes need to make other decisions as well.

Finally go through this system, your diagram and list of events, and make sure every wait and unlock adds up. Make sure no thread is starved and that no locks cause deadlocks, but also that no thread can get through the system without waiting for the appropriate timing.

Once you have one such system described you can add the events need to solve the next level of complexity, and go through the steps again for that system.

Have a good time!