Uppgift 6, Miniprojekt

Ni får välja en av 3 projektalternativ. Förslag finns nedan.

Allmänt om projektet och redovisningen

Ni får arbeta i grupp, 4 studenter. I princip skall du arbeta med din labbgrupp men om han/hon inte följer kursen längre får ni omgruppera. Du får inte göra den sista uppgiften om du inte har mist 4 av 5 labbar godkända. Du får göra först dina labbar.
Hör av dig till mig om du saknar labbpartner.

Ni skall redovisa lösningen i en muntlig presentation. Blir inte presentationen godkänt eller om du inte deltar i redovisningen den 19/5 då måste du redovisa projektet med skriftligt rapport. Se mall för skriftlig rapport.

Följande gäller för muntliga presentationer.
En ppt. med 5-6 sidor som skall innehålla.

1. Problem formulering
2. Krav. Vilka krav skall programmet uppfylla?
3. Översikt över lösningen. Systembeskrivning(se exempel i mall för rapport).
4. Algoritmbeskrivning för någon/några av viktigaste algoritmerna (se exempel i mall för rapport).
5. Resultat. Visa att kraven är uppfyllda och hur den har testats.

Exekvera och visa hur programmet fungerar

Följande gäller för skriftliga presentationer.

Se bifogad mal för rapport. Det är viktigt att du beskriver din kod men klassdiagram samt de viktigaste algoritmerna men pseudokod. Rapporten skall vara i pdf format max 5 sidor (utan källkod). Kontakta gärna mig om ni har frågor om uppgiften.

Följande gäller för koden

Koden skall vara kommenterad med java taggar och indenterad. Identifierare för klasser, variabler och metoder väl valda. Du skall följa java standard.

Är du inte klar med uppgiften tills den 19/5 får du redovisa efter sommaren under omtentavecka. Koden och rapporten skall mailas till mig den 15/8.

Val 1. Webbaserad Akutmottagning

Internät tjänster använder olika typer av kö system. För en online akutmottagning behövs ett system som ”kallar” in patienter både utifrån deras könummer men också utifrån deras ”emergency number” om det finns. ”Emergency numbers” finns bara i intervallet 1(akut)-2(alvarligt) -3(mycket alvarligt). Detta är en prioritet som patienten får utifrån t.ex vilken symptom man anger.

I din uppgift skall du implementera en Prioritetskö där patienternas läggs till när de inskrivs till akutmottagningen samt tas bort när de har blivit behandlade. Du skall också kunna ändra prioritering för en patient efter det att den har lagts i kön (i fall att patienten blir sämre under tiden den väntar).

Alltså högsta prioritet har patienten med ”emergency numbers” 3. Om ingen patient i kö har ”emergency numbers” (1-3) då behandlas vanliga patienter i den ordningen de anländer till akuten(könummer ordning).

Implementera först klassen ”Patient” och de datastrukturer som du behöver- När din datastruktur är klar, implementera ett simuleringsprogram där patienter läggs i kön, ta patienter bort från kön, ändrar prioritet, mm.

Du kan också bestämma en max storlek för din kö. Om storleken blir större än så skall flera läkare ”kallas” in…(t.ex).

Visualisera på något sett hur kön utvecklas. Visualisera gärna med en GUI. Gör en simulering. Bestäm några parametrar som kan påverka scenariot och ”skruva på de parametrarna för att se hur programmet beter sig”

Använd en egen Heap representation för din Prioritetskö.

Val 2. Musikspelarverktyg

I ett musikspelarverktyg vill du effektivt bygga musiklistor och söka både utifrån artistens namn men också kunna söka efter låtar.
Du har ljudfiler som du kan söka igenom på det mest effektivaste sättet (kanske en hashtabell). Du skall kunna skapa andra listor sorterade efter artist eller låtnamn, mm.

Ett förslag finns beskrivet nedan men det är också fritt för dig att hitta en annan representation.

Ditt spelarverktyg liknar lagerprogrammet från lab3 men det är implementerat med en hashtabell istället för länkad lista. Din Item-klass skall innehålla uppgifter om låtnamn, artist, ljudfilens namn, låtens speltid, annat om du behöver.

Du kan göra en implementering av en öppen hashtabell . Men för varje position i tabbelen skall du koppla ett sökträd som innehåller alla artistenslåtar. De träden representeras med en egen binärt sökträd som du konstruerar från grunden och där du implementerar bara de operationer du behöver. Se till att trädet är balanserad.

Varje träd i tabellen skall innehålla alla objekt (låt) som tillhör samma artist.
Det kan vara lämpligt att implementera en sak i taget och testa.

För detta skall du använda en eget skriven träddatastruktur. Kalla den ItemTree och representeras som en enkel sökträd-datastruktur. De metoderna du behöver är insert(), makeEmpty() och printInOrder(). Inspirera dig från vilket balanserat träd datastruktur du vill.

Du skall också kunna skriva ut bara de listor som tillhör samma nyckel showlist(String K) (där K är nyckel alltså namn på artist). Listan skall vara sorterad efter låtens namn.

Du har fria händer att designa dina datastrukturer och applikation. Använd dina egna ljudfiler som du har på datorn.

Gör en enkel GUI för ditt verktyg där du kan också välja och spela musikfiler.

Val 3. Trafik kontrollsystem

Den förarlösa bilen kommer i framtiden att få information från trafikkontroll system i visa situationer.
Skriv ett java program som simulerar en trafik kontrollsystem för en 4 väg korsning. Systemet kontrollerar 4 trafiksignal, kalla de nord, syd, est, väst. För varje trafiksignal gäller, röd, gul och grön ljus.
Systemet skall:
-Aldrig ge grön signal samtidigt för trafik som kan kollidera
-Undvika trafik stockningar
-Ge automatisk grön ljus till ”emergency” bilar

Implementera först lämpliga datastrukturer för programmet.
Använd trådar för att kunna simulera trafik som kör samtidigt
Använd prioritesköer för ”emergency ” bilar. Tänk att detta kan inträffas i varje riktning
Utveckla en interface som kan visa simuleringen av trafiken. Gärna GUI.