Godkendelsesopgave 6 i "Styresystemer og multiprogrammering"

Generelt

Denne ugeopgave stilles fredag den 18. marts 2011 og skal afleveres senest *fredag* den 21. marts 2011 klokken 23:59:59. Bemærk at dette betyder at der er kortere tid til at løse denne godkendelsesopgave. Det er *ikke* muligt at gen-aflevere denne godkendelsesopgave. Godkendelsesopgaven kan løses af grupper på op til 2 personer (3 kan tillades, men frarådes). Besvarelsen af opgaven vil resultere i enten 0, 1/2 eller 1 point. Pointene uddeles efter følgende retningslinjer:

- 0 point: besvarelsen har flere store mangler.
- 1/2 point: besvarelsen opfylder i store træk kravene, men har flere mindre mangler.
- 1 point: besvarelsen opfylder kravene til opgaven med kun få eller ingen mangler.

Det er en betingelse for at gå til eksamen på kurset at man har opnået mindst 4 point i alt.

Besvarelsen skal indleveres elektronisk via kursushjemmesiden (Absalon). Besvarelsen bør ske ved aflevering af en enkelt fil. Brug 'zip' eller 'tar.gz' til at samle flere filer. Opgavenummer og navne på gruppemedlemmer skal fremgå tydelig af første side i besvarelsen. Når I indleverer bør efternavne på alle gruppemedlemmer indgå i filnavnet - desuden skal opgavenummer fremgå af navnet:

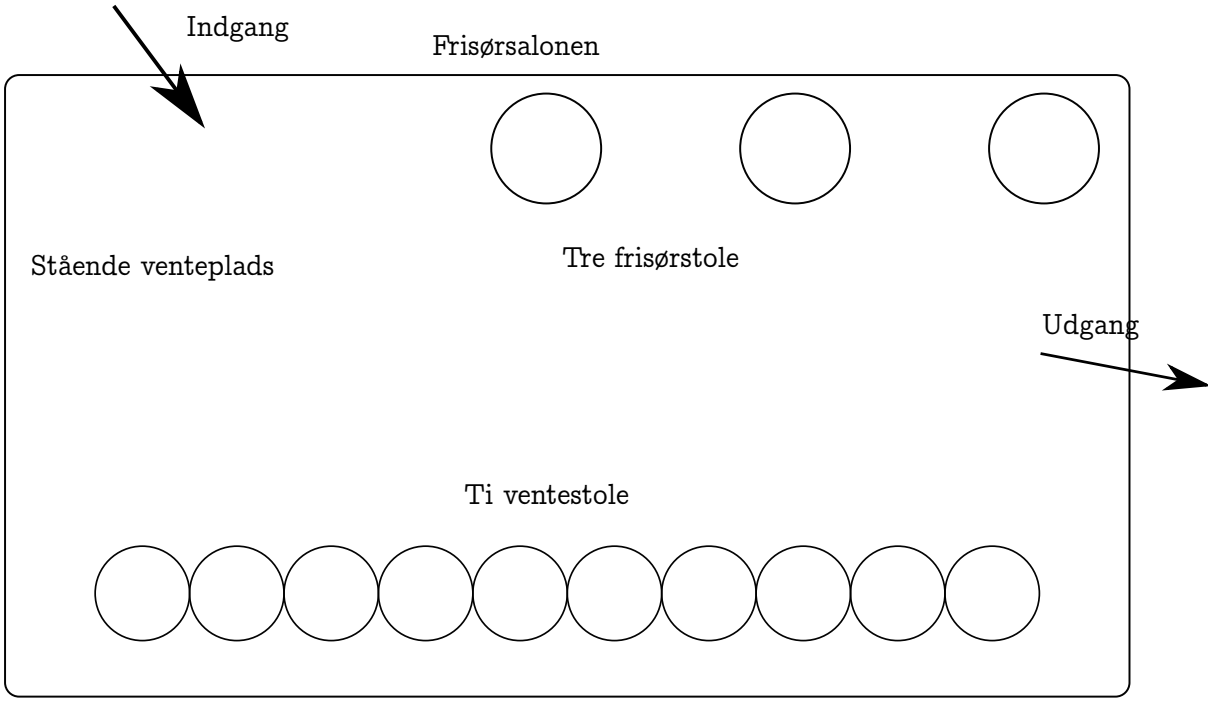
```
efternavn1-efternavn2-efternavn3-G1.<endelse>
```

Jeres aflevering skal være i et format der kan læses på DIKUs systemer uden problemer (således bør formatet f.eks. ikke være MS Word dokumenter). Se i øvrigt "Krav til G-opgaver" siden på kursushjemmesiden under "G-opgaver" menupunktet.

I skal kun aflevere én fuld besvarelse per gruppe. Personen, som afleverer skal markere sine gruppemedlemmer via Absalon.

Om dokumentation af jeres løsning

Afleveringen skal indeholde en kortfattet rapport der dokumenterer hvilke vigtige observationer I har gjort jer, antagelser jeres løsninger afhænger af (som I enten har valgt, eller som er valgt af designerne af BUENOS) og desuden skal I redegøre kortfattet for designbeslutninger i den kode I har implementeret. I skal også huske at kommentere jeres kildetekst så den er let at forstå. Især er det vigtigt at dokumentere hvis I har foretaget ændringer i allerede implementerede dele af BUENOS.



Figur 1: Overblik over frisørsalonen

1 Denne uges tema: Klassisk multiprogrammeringsøvelse

I denne godkendelsesopgave kan I benytte jeres modificerede BUENOS kerne fra G3 som inkluderer låse og betingelsesvariable samt brugertråde. Hvis I ikke stoler på jeres egen implementation, kan I også benytte den udleverede referenceløsning til G3.

G6.1: Frisørsalonen

I denne opgave skal I implementere en hypotetisk frisørsalonsimulering. Nedenfor følger en beskrivelse af problemstillingen, som skal simuleres.

Tre frisører arbejder uafhængigt af hinanden i en frisørsalon (se figur 1):

- Frisørsalonen har 3 frisørstole og ved hver stol arbejder netop én frisør (og frisører skifter ikke stole).
- Hver frisør arbejder på samme måde:
  - Frisøren sover (eller dagdrømmer) når der ikke er nogen kunder, som venter og der ikke allerede sidder en kunde i frisørens frisørstol.
  - Når frisøren sover, venter han på at blive vækket af en kunde. Et skilt i salonen angiver hvilken frisør, som har sovet længst tid. En kunde, som skal klippes, vækker den frisør, der har sovet længst tid.
  - Når en frisør er blevet vækket af en kunde, sætter kunden sig i frisørstolen og frisøren klipper kundens hår.
  - Når frisøren er færdig med at klippe en kunde, betaler kunden og går sin vej.
  - Når en frisør har modtaget betaling, kontrollerer frisøren om der er ventende kunder i en af ventestolene. Hvis der er en ventende kunde, flyttes kunden til frisørstolen og

kundens hår klippes. Hvis der ikke er nogen ventende kunder, falder frisøren i søvn og venter på at blive vækket af en kunde.

- Hver kunde opfører sig efter følgende regler:
  - Når en kunder kommer ind i frisørsalonen kontrollerer kunden hvor mange kunder, der venter. Hvis flere end 20 andre kunder venter, forlader kunden salonen igen uden at blive klippet.
  - Dernæst undersøger kunden om der er ventetid på at blive klippet. Hvis mindst én frisør sover, vækker kunden den frisør, som har sovet længst tid og sætter sig i frisørens stol (og bliver dermed klippet).
  - Hvis alle frisører er optaget af at klippe andre kunder, venter kunden. Hvis der er en tom ventestol, sætter kunden sig i denne. Ellers står kunden op og venter indtil en ventestol bliver tom (hvorefter kunden sætter sig til at vente i denne).
  - Kunder holder styr på deres ventetid således at det altid er den kunde, som har ventet længst tid, som bliver valgt til at blive klippet som den næste.
  - Tilsvarende holder kunder styr på, hvem som har ventet på en ventestol i længest tid. Den kunde, som har ventet i længst tid på en ventestol er den kunde som får den næste tomme ventestol.

Opgave 1: Implementer en frisørsalonsimulering. I skal implementere et (multi)program, som simulerer hvordan kunder og frisører handler over tid.

- Simuler hver frisør og kunde som separate processer.
- Minimum 30 kunder skal forsøge at blive klippet (men I behøver ikke garantere at alle kunder bliver klippet).
- I skal simulere at klipning af en kunde tager tid.
- Hver frisør skal som rapportere (rapportering kan ske ved udskrivning til konsol eller en fil via "syscall_write") når klipning påbegyndes og når klipning afsluttes.
- Hver kunde skal rapportere når han/hun ankommer til frisørsalonen. Kunden skal ligeledes rapportere om salonen er fyldt og kunden derfor forlader salonen uden at blive klippet.
- Hver kunde skal rapportere når han/hun venter og når ventetiden er slut; både for stående ventetid og for ventetid i en ventestol.

I ovenstående er alle rapporteringskrav *minimumskrav*. Det er tilladt at frisører og kunder rapporterer mere end nævnt. Et eksempel uddata fra en kørsel af en simulation kunne således være:

```
== Frisørsalonsimulering start ==
Frisør[0]: sover
Frisør[1]: sover
Frisør[2]: sover
Kunde[0] : ankommer til salon (som ikke er fyldt)
Kunde[0] : vækker frisør[0]
Frisør[0]: klipper kunde[0]
Kunde[1] : ankommer til salon (som ikke er fyldt)
```

```
...
Kunde[3] : ankommer til salon (som ikke er fyldt)
Kunde[3] : alle frisører er optaget; venter
Kunde[3] : indtager ledig ventestol 0
Frisør[2]: færdig med at klippe kunde
Kunde[2] : forlader salon nyklippet
Frisør[2]: ser efter ventende kunder
...
```

Det er tilladt at simulere at klipning tager tid med en "busy-wait" løkke:

```
void simulatedwait(int duration) {
  int loc;
  for(loc = 0; loc < duration; loc++) {
    // no-op
  }
}
```

Figur 2: Betragt dette som et "easter-egg"