Øvelse 4 – Synkronisering av prosesser/tråder – Hele øvelsen er obligatorisk

Dette er en øvelse hvor alle oppgavene er obligatoriske med innlevering. Hjelp til øvingen gis på laben i ukene 45 til 47.

Krav til den obligatoriske øvelsen:

- Dere kan arbeide sammen i grupper på inntil 4 personer.
- Gruppen leverer samlet på Canvas.
- Innleveringsfrist er 24. november.
- Dere må løse alle oppgavene.
- Dere må levere: Dokument (f.eks. PDF, tekstfil, men ikke noe proprietært filformat) med løsninger på oppgavene. Kildekode, fil som beskriver hvem som eventuelt har jobbet sammen (med navn README), skjermskudd som viser kjøring av programmene.

Innhold

Oppgave 1

a) Hva er et operativsystem?
b) Hva er en prosess?
c) Hva er forskjellen mellom en tråd og en prosess?
d) Hva er en kritisk region (critical section)?
e) Hva er en semafor?
f) Hvordan kan vi implementere en kritisk region ved semaforer? Skriv en liten algoritme enten i Java, eller på norsk som en algoritme. Kall den kritiske regionen for "kritisk region" uansett algoritmespråk. Anta også at du har datatypen (klassen) Semaphore tilgjengelig med metodene wait og signal.
g) Gi en kort beskrivelse av "banksjefens algoritme".
h) Vi skal se på 4 prosesser P_1, P_2, P_3, P_4 og deres bruk av ressurstypene A, B og C. Vi har følgende tilstand:

	Allokert	**Maks behov**	**Tilgjengelig**
	A B C	A B C	A B C
			2 3 0
P_1	0 1 0	7 5 3	
P_2	3 0 2	3 2 2	
P_3	3 0 2	9 0 2	
P_4	1 1 2	4 3 3	

i. Vis at systemet er i en sikker tilstand.
ii. Kan forespørselen (2 1 0) fra P4 godkjennes? Begrunn!
iii. Kan forespørselen (1 1 0) fra P1 godkjennes? Begrunn!

Oppgave 2

Lag en løsning i Java på problemstillingen "bounded-buffer" med Java sin Semaphore-klasse.

Oppgave 3

Lag en løsning i Java på problemstillingen "readers-writers" med Java sin Semaphore-klasse.

Oppgave 4

Lag en løsning i Java på problemstillingen "the dining-philosophers" med Java sin Semaphore-klasse.