For å få til dette producer/consumers problemet, har jeg brukt flere ting i java-systemet. For det første har jeg brukt synkroniserte metoder. Når metodene i en klasse er synkronisert, får man en monitor. Det vil si at bare en tråd av gangen kan kjøre en av metodene. Altså når en tråd kjører en av de synkroniserte metodene, vil alle andre tråder som prøver å kjøre en synkronisert metode på det objektet, blokke. Det vil si at de vil stoppe å kjøre frem til det er deres tur.
Jeg har også brukt noe som heter wait og notify. Når en tråd kommer til en wait inne i en synkronisert metode, vil den slippe låsen, så andre metoder kan kjøre inne i de synkroniserte metodene. Den vil så blokke, og vente på at noen kaller notify inne i en av de synkroniserte metodene. Når noen så kaller notify, våkner den opp igjen, men kan naturligvis ikke begynne å kjøre før den tråden som kalte notify er ute av den synkroniserte metoden.
Det fine er også at hver gang man kaller notify, vil en og bare en tråd, våkne. Så hvis flere tråder blokker, for eksempel når alle frisørene venter på en kunde, vil bare en av frisørene starte igjen når innkasteren kaller notify.