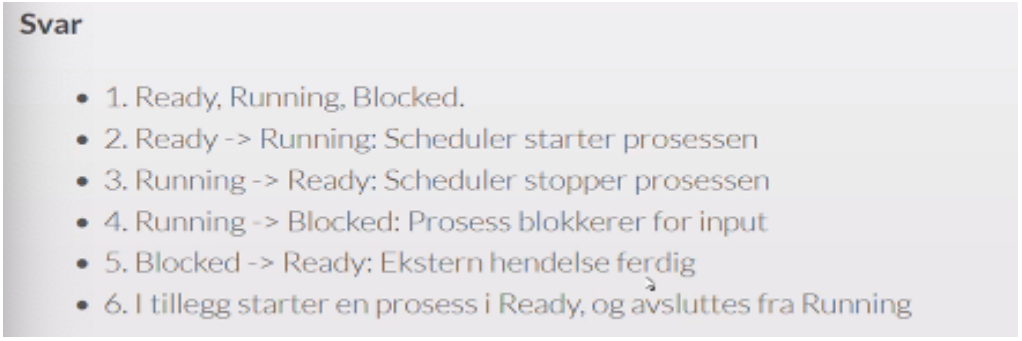
Hva er forskjellen på et program og en prosess?

Et program er en fil med executable-format
En prosess er en kjørende instans av et program

Hvilken prosess-tilstander har vi, hvordan beveger man seg mellom de forskjellige tilstandene?

Disse er følgende prosess-tilstander vi har tilgjengelig: Running, Ready, Blocked
Vi kan bevege oss fra følgende tilstander som kan vises gjennom bildet under:



Hva er en context-switch? Hvilke steg foretar kjernen i en context switch?

En context-switch er prosessen av å bytte mellom en kjørende prosess. Vi kan ha to tasks som vi kan kalle “A” og “B”, hvor om “A” er en kjørende tilstand, men så skal “B” få kjøre sin prosess. Det som skjer, er at vi stopper den kjørende tilstanden “A”, lagre tilstanden til den prosessen også laster inn neste prosess sin tilstand. Tilstanden kan pushes i stacken eller i PBC(Program control block), men OS har også en betydning. Tilslutt kan vi starte neste neste prosess.

Hvorfor er det umulig å lage en scheduleringsalgoritme som passer for alle systemer og formål?

Det er en rekke ting vi må ta hensyn til f.eks, hvilke prosesser er det vi tar inn, er de CPU-drevne prosesser eller I/O bundne-prosesser. Har vi en skedulerer som oppfører seg som en “preemptiv-skedulering” eller “ikke preemptiv-skedulering”. Derfor så er det ikke akkurat en beste mulig algoritme ettersom forskjellige systemer har forskjellige formål hvor et gitt skeduleringsalgoritme vil være den mest optimale for dette systemet.

Algoritmen påvirker fairness, starvation, response time, throughput, utilization, overhead, predictability (og mer?). Forskjellige systemer har forskjellige formål og dermed forskjellige krav til målene over. Typiske systemer: Batch, interaktive, real-time osv.

Beskriv hva som skjer når en bruker gjør et tastetrykk. Svarer systemet forskjellig for preemptive eller ikke-preemptive systemer? Hvorfor eller hvorfor ikke?

Ved et tastetrykk så vil kastes ut en “Interrupt” som CPU håndterer. Tastetrykk går til prosessen som er i fokus, men ikke nødvendigvis den som kjører. Preemptive systemer vil kunne stoppe kjørende prosess og håndtere tastetrykket. Gir mer overhead, men raskere responstid. Non-preemptive vil vente til kjørende prosess er ferdig og så håndtere tastetrykk. Gir lite overhead, men treg og varierende responstid.

Når kjøres en scheduleringsalgoritme?

- Når en prosess lagres
- Når en prosess terminerer
- Prosess blokkerer
- For preemptive systemer som vil interrupt være aktuelt
- Klokke-interrupt (preemptivt system), hvis du husker “Stein” sin klokke-algoritme når det gjaldt tråder, så vil tråder satt på vent, ha en klokke som angir hvor lenge de skal vente. Dette fungerer likt som ved klokke-interrupt i et (preemptivt system).