Laboration 3: Trådprogrammering

Uppgift 1: Simulering av de dinerande filosofernas problem
Skapa en simulering av The Dining Philosoper's Problem med tidsskalan 900:1. (En körrunda blir då 48 sekunder.) Ättiden är ett slumptal mellan 16 och 45 minuter (simulerad tid) (1-3 sekunder i realtid) och tänktiden är ett slumptal mellan 30 och 90 (simulerad tid) (2-6 sekunder i realtid). Ditt program mottar en kommandoradsparameter som är antalet filosofer. Programmet ska skrivas i *C* med biblioteket `pthread`. Under *Debian Linux* kompilerar man ett ptrådsprogram med `gcc program.c -lpthread -o program`, där `program.c` är källkoden och `program` är det man vill att den körbara filen ska heta.

• Ditt program ska vara baserat på ett skelett som finns på kurswebben. Detta är för att underlätta redovisning och jämförelse mellan olika lösningar.
• Programmets beteende ska styras med kommandoradsparametrar och en av parametrarna ska vara antalet filosofer och den andra ska ange om programmt säkert ska skapa en låsning eller inte. Vi ska studera begreppet låsning genom att skriva ett program som **säkert skapar låsning** eller som **säkert undviker låsning**, då klargörs skillnaden men även orsaken till låsning. För att säkert skapa låsning kan man inte inkludera begreppet "svält" som nämnts på föreläsningen – alla filosofer måste vara odödliga. Filosoferna ska vara helt oberoende av varandra, det vill säga det ska finnas N st likadana parallella trådar som ska synkroniseras vad gäller användningen av bestick. Man får absolut inte bara ha en enda mutex för alla bestick, man måste ha en array av mutexar, en mutex för varje bestick.

Redovisning

Muntlig redovisning framför datorn samt inlämning för plagiatkontroll på samma sätt som laboration 2: alltså på `https://maceo.sth.kth.se`. Du ska vid laborationens slut kunna redogöra för och visa upp en fungerande lösning för respektive moment. Slutförandet av laborationen ska bestå av

• Redovisning av hur du har gått tillväga för att lösa laborationen (både kommandon och, i förkommande fall, resultat)
• Vad du anser att du har lärt tid av denna laboration
• Redovisning av nedlagd tid på laborationen.

Man får inte arbeta två och två med denna uppgift, däremot måste man (så långt det går) redovisa två och två.

Extra icke-obligatorisk uppgift: (För er som vill ha något att bita i, kan också vara bra att studera inför tentamen): Skriv om laboration 2 till en chat som är flertrådad istället för en kommandoserver som är flerprocessig och som använder pipes lokalt för att skicka alla repliker till alla servertrådar och servertrådarna i sin tur skickar alla repliker via sockets till alla klienter. Man ska kunna kliva ur chatten genom att skriva /quit och när man kommer in i chatten får man en fråga om vad man vill kalla sig. Vissa begränsningar kan behövas... vilka? Minns att när man använder en pipe lokalt i ett program som inte blir flera processer så slipper vi problematiken med dubbla läs- och skrivändar... Detta är en avancerad extrauppgift som också ges i en version i årskurs 3 då man använder Javas Remote Method Invocation – mer om det om cirka 1.5 år. (Om allt går enligt planerna.)