LUNDS TEKNISKA HÖGSKOLA
Datavetenskap

EDA095 FED

Tentamensskrivning — Nätverksprogrammering 2004-05-25, kl 8-13

DEL 1 - Frågor av teoretisk, principiell eller utredande karaktär

Anvisningar

Tillåtna hjälpmedel för denna del av tentamen: **Inga**. Kurslitteratur och andra hjälpmedel för del 2 av tentamen skall förvaras på golvet bredvid bordet eller vid salens vägg.

Denna tentamen i kursen Nätverksprogrammering består av två delar - en del som innehåller frågor av teoretisk/principiell/utredande karaktär och en del som innehåller praktiska programmeringsuppgifter. Detta är del 1. När du löst uppgifterna i denna del av tentamen lämnar du in din lösning i det vita tentamensomslaget varvid du erhåller del 2 av tentamen tillsammans med ett nytt, färgat, tentamensomslag som skall användas vid inlämning av din lösning på del 2 av tentamen.

För godkänt betyg på tentamen krävs sammanlagt minst 20 poäng på tentamen, varav minst 8 poäng på vardera deltentamen. För högre betyg krävs naturligtvis mer, så gör så många uppgifter du kan.

Senast 2004-05-28 anslås på institutionens anslagstavla vilka som deltagit i tentamen men som, enligt institutionens noteringar, ännu inte redovisat laborationerna eller projektet. Deras skrivning rättas inte (och kan på sikt komma att annulleras) såvida inte rättelse skett eller dispens erhållits från ansvarig lärare. Rättelse skall göras senast 2004-06-11.

Uppgifter

1. **a)** Vad står följande två förkortningar för:
1. UDP
2. TCP (1p)

b) Jämför UDP med TCP. Hur skiljer de sig åt? Vilka egenskaper har UDP som inte TCP har och omvänt? (2p)

2. Vad är Common Gateway Interface? Beskriv kortfattat hur det fungerar. (2p)

3. I Internetsammanhang pratar man ofta om så kallade RFC-er som dessutom är numrerade, t.ex. RFC768. Vad står förkortningen RFC för och vad är en RFC? (1p)

4. Beskriv kortfattat vad som händer i en webbserver när en begäran anländer från en klient om att visa en viss sida som är implementerad med hjälp av JSP - Java Server Pages. Dvs, ange de olika stegen i processen från det att en "GET fil.jsp"-begäran anländer till det att den resulterande HTML-koden sänds till klienten. (2p)

5. I samband med RMI talar man om *skelett* (skeletons) och *stubbar* (stubs). Vad innebär dessa begrepp och på vilken sida av nätverksförbindelsen hittar man respektive fenomen? (2p)

6. Para ihop var och en av följande fyra beskrivningar med en av de nedan angivna standardklasserna i Java.

a) Ström som används för att lagra data på ett skivminne.
b) Superklass för alla klasser som representerar utgående strömmar.
c) Superklass för flera av de klasser som utökar funktionaliteten hos en existerande ström.
d) Innehåller metoder för att skicka olika typer av primitiva datatyper, t.ex. heltal, flyttal.

Klasser
DataOutputStream, CipherOutputStream, ObjectOutputStream, FileOutputStream, BufferedOutputStream, PrintStream, PrintWriter, FilterOutputStream, OutputStream.

(2p)

7. En *binär semafor* är en mekanism som ibland används för att åstadkomma ömsesidig uteslutning i ett program. Den kan beskrivas som ett objekt som har två metoder: `take()` och `give()`. När ömsesidig uteslutning önskas anropar en tråd metoden `take()` och när ömsesidig uteslutning inte längre är nödvändig anropar tråden `give()`. Om någon annan tråd försöker anropa `take()` under denna tid kommer den att blockera inuti anropet av `take()` tills dess att `give()` anropats av den första tråden.

Programmeraren JavaJeppe har skrivit följande klass som implementerar en binär semafor:

```
public class BinarySemaphore {
    boolean locked = false;

    public void take() {
        while(locked) {
            // Do nothing - just wait for locked to be false
        }
        locked = true;
    }

    public void give() {
        locked = false;
    }
}
```

a) Vad innebär begreppet ömsesidig uteslutning (mutual exclusion)? (1p)

b) Vad brukar man kalla den i `take()` ovan använda tekniken för att vänta på ett villkor? (1p)

c) Denna metod har en stor nackdel som gör att den oftast bör undvikas. Vilken? (1p)

d) Förutom problemet som söktes i uppgift c) lider lösning dessutom av problem med så kallad *kapplöpning* (race condition). Om vi har en väldig otur med hur systemet väljer att schemalägga de olika trådarna finns det en risk för att flera trådar släpps förbi anropet av `take()` samtidigt.

Skriv om klassen `BinarySemaphore` ovan så att nackdelen i uppgift c) elimineras helt tillsammans med risken för kapplöpning. Använd Javas monitorbegrepp för att åstadkomma detta. Klassen ska ur de anropande trådarnas synvinkel fungera likadant som tidigare. (3p)

8. Vilka av följande fyra påståenden är korrekta?

a) UDP är ett säkrare överföringsprotokoll än TCP därför att UDP kontrollerar att inga meddelandepaket förvanskas eller försvinner på nätverket och skickar om nödvändigt om meddelandepaketen.

b) Ett objekt av klassen `DatagramSocket` kan användas för att skicka meddelanden till flera olika mottagare omväxlande och kan även användas för att samtidigt lyssna efter, och i tur och ordning ta emot, meddelanden från flera olika avsändare.

c) Om man deklarerar en tråds `run()`-metod synchronized i Java innebär det att tråden kommer att köra `run()`-metoden regelbundet.

d) Om ett program lyssnar efter UDP-paket på ett visst portnummer kan ett annat program på samma dator starta en TCP-server som accepterar TCP-uppkopplingar på samma portnummer.

(2p)
