

Oppgave 2 Datakommunikasjon (ca. 35 %)

Svar *kort og presist* på følgende oppgaver om datakommunikasjon.

2a Internet

Beskriv kort hva vi forstår med Internet.

2b Protokoll

Hva forstår vi med en protokoll i datanett-sammenheng?

2c Referansemodeller

Hvorfor har man laget såkalte “referansemodeller” for datakommunikasjons-systemer, hva slags struktur har disse modellene og hvilke hovedfunksjoner er beskrevet? Beskriv likheter og forskjeller mellom ISOs OSI modell og Internets TCP/IP modell.

2d Overføringsmedia

Hvilke fysiske overføringsmedia brukes vanligvis i datakommunikasjons-sammenheng? Beskriv kort fordeler og ulemper ved de ulike mediene.

2e Svitsjeteknikker

Gjør rede for forskjellen mellom linjesvitsjing og pakkesvitsjing. Hva er fordeler og ulemper med de to teknikkene?

2f Forbindelsorientert vs. Forbindelsesløs

Hva slags funksjonalitet ytes av henholdsvis forbindelsorienterte og forbindelsesløse tjenester for dataoverføring? Beskriv fordeler og ulemper ved disse to tjenestekategoriene.

2g Multipleksing

Hvilke tre hovedteknikker for multipleksing over et delt overføringsmedium benytter man? Beskriv disse.

2h IP adressestruktur

Gjør rede for hvilke to måter man vanligvis bruker for å angi en IP-adresse og forklar hvorfor man har valgt en slik adressestruktur.

2i Ruter

Hvilke funksjonelle komponenter finner man i en ruter?

2j Overførings-syntaks

Hva menes med en overførings-syntaks og hvorfor trenger man en slik spesifisering?

(Fortsettes på side 3.)

Oppgave 3 Diverse flervalgsoppgaver (ca. 30 %)

I denne oppgaven skal du liste opp alle påstander som er SANNE uten å gi noen forklaring. Merk dere at flere kan være riktig i samme oppgave. Hvert riktig svar gir poeng, og hvert gale svar gir minus-poeng (poeng trekkes).

3a C programmering

Anta at du har en funksjon med koden beskrevet i figur 1 og denne kompiles med `gcc` som dere har brukt i kurset. Hvilke utsagn er riktige:

1. En program-fil (`funcA.c`) som inneholder denne funksjonen vil ikke kompilere.
2. På linje 4 vil verdien 1000 bli satt inn i `a[0]` og 100 bli satt inn i `a[1]`.
3. På linje 4 vil verdien 1000 bli satt inn i `a[1]` og 100 bli satt inn i `a[0]`.
4. `for`-løkken på linje 13 vil bli eksekvert fem ganger.
5. `for`-løkken på linje 13 vil aldri bli eksekvert.
6. Startverdien til `a` i `for`-løkken på linje 13 er 100.
7. `printf`-linjen på linje 9 vil skrive ut tallet "101".
8. `printf`-linjen på linje 9 vil skrive ut tallet som samsvarer med adressen til `a[0]` i minne.
9. `if`-betingelsen på linje 12 er alltid FALSE.
10. `printf`-linjen på linje 10 vil skrive ut teksten "1001 1000".
11. `printf`-linjen på linje 10 vil skrive ut teksten "1001 1001".
12. Deklarasjonen av `int a[2]` på linje 13 er tillatt.
13. Etter deklarasjonen av `a` på linje 13 er både `a[0]` og `a[1]` lik 0.
14. I uttrykket på linje 16 vil alltid 1000 bli subtrahert fra `b`.
15. Verdien til `a[1]` på linje 16 er forskjellig for hver iterasjon av `for`-løkken.
16. `printf`-linjene på linje 18 og linje 20 vil skrive ut den samme teksten.
17. `printf`-linjen på linje 18 vil skrive ut teksten "500 400 1".
18. `printf`-linjen på linje 18 vil skrive ut teksten "100 1001 1".
19. `printf`-linjen på linje 18 vil skrive ut teksten "500 500 501".
20. `printf`-linjen på linje 20 vil skrive ut teksten "500 400 1".
21. `printf`-linjen på linje 20 vil skrive ut teksten "100 1001 1".
22. `printf`-linjen på linje 20 vil skrive ut teksten "500 500 501".
23. Funksjonen har ingen retur-verdi.

(Fortsettes på side 4.)

```
1: void A(void)
2: {
3:     int *p, b;
4:     int a[2] = {100, 1000};
5:
6:     p = a;
7:     b = ++(*(p + 1));
8:
9:     printf("%d ", *p);
10:    printf("%d %d\n", a[1], b);
11:
12:    if (*a == 100) {
13:        int a[2];
14:        for (*a = *p ; *a < 500 ; *a += 100) {
15:            *(a + 1) = a[0];
16:            b = b - a[1];
17:        }
18:        printf("%d %d %d\n", *a, a[1], b);
19:    }
20:    printf("%d %d %d\n", *a, a[1], b);
21: }
```

Figur 1: Funksjonen A

3b Operativsystemer

Hvilke utsagn er riktige:

1. Vanlige brukerprogrammer har vanligvis full aksess til alle ressurser og instruksjoner på en datamaskin.
2. En "scheduler" bestemmer hvilken jobb som skal få bruke en gitt ressurs - det vil si at den bestemmer i hvilken rekkefølge forespørslene blir utført.
3. Forskjellen mellom en prosess og et program er at en prosess er binærfilen (den kjørbare filen) som genereres når man kompilerer en programfil skrevet i C (en tekstfil).
4. En sidefeil (page fault) finner sted hver gang vi aksesserer operativsystemkjernen.
5. Sideutbyttingsalgoritmen (paging algoritmen) *least recently used* (LRU) bytter alltid ut den minnesiden som ble aksessert først.
6. Systemer som Windows og Linux bruker ikke virtuelt minne.
7. Formålet med en buffer-cache er å redusere antall diskaksesser.
8. En *inode* inneholder metadata ("data om data") for en fil som for eksempel filnavn, eier og pekere til datablokker.
9. En *pipe* er en bufferstruktur for å holde minnesegmenter med fast størrelse i minnet.

3c Datakommunikasjon

Hvilke utsagn er riktige:

1. Et aksessnettverk er et nettverk som kontrollerer aksessrettigheter.
2. Kjernenettverket i Internet er det som kople aksessnettene sammen.
3. Et endesystem er en maskin eller et program som befinner seg i den ene enden av en nettforbindelse.
4. TCP er en protokoll som nettverket bruker for at datapakkene skal finne fram til riktig maskin.
5. Oppgaven til nettlaget er å kople de fysiske lagene sammen til ett fysisk nettverk.
6. De lagene som er aktive i en ruter er det fysiske laget, linklaget og nettlaget.
7. Med stadig raskere linkteknologi får vi overført mer data per sekund, men reisetiden for dataene på linken forblir omtrent den samme.
8. En web-tjener vedlikeholder ikke tilstandsinformasjon om tidligere klient-kall.
9. Flytkontroll er en linklagsfunksjon som kontrollerer dataflyten på en fysisk link.