Eksamen i fag
SIF8010 Algoritmer og Datastrukturer
Mandag 17. Desember 2001, kl 0900-1500

Faglig kontakt under eksamen: Arne Halaas, tlf. 73 593442.
Hjelpemidler: Alle kalkulatortyper tillatt. Alle trykte og håndskrevne hjelpemidler tillatt.
VIKTIG: Alle svar skal skrives inn på vedlagt Svarark, og kun der. Før på "Student.nr."
NB.: Oppgavene er utformet slik at de eliminerer svarmuligheter som kan gi grunnlag for klaging.

Oppgave 1.

a) Hvilket uttrykk er det som best beskriver klassen av funksjoner der en blant annet finner løsningen til rekurrensligningen T(n) = 3T(n-9) + 1, der T(0) = 1 ?

1: logaritmisk, 2: lineær, 3: kvadratisk, 4: kubisk, 5:eksponensiell, 6: Ingen av alternativene.

b) Hvilken av de nedenstående rekurrensligning beskriver best en splitt-og-hersk-algoritme som med c*n operasjoner deler et problem i fire deler som hver er en tredjedel i størrelse av hovedproblemet, som deretter løser underproblemene rekursivt og til slutt kombinerer del-løsningene ved hjelp av $d*n^2$ operasjoner?

1: $T(n) = T(4n/3) + n(c+dn)$, 2: $T(n) = 4T(n/3) + n^2(c/n+d)$, 3: $T(n) = T(4n/3) + cdn^3$
4: $T(n) = 4T(n/3) + cdn^3$, 5: $T(n) = 4T(n)/3 + cdn^3$, 6: $T(n) = 3T(n/4)+ cn + dn^2$,
7: Ingen av alternativene 1-6.

c) Løsningen for rekurrensligningen T(1) = c; T(n) = T(n/2) + O(1), n > 1, er T(n) =

1: $\theta(\log n)$, 2: $\theta(n)$, 3: $\theta(n \log n)$, 4: $\theta(n^{c/2})$, 5: $\theta(n^c \log n)$, 6: Ingen av alternativene 1-5.

d) Hva er løsningen på rekurrensligningen T(1) = 1; $T(n) = T(n^{1/2}) + O(1)$, n > 1?

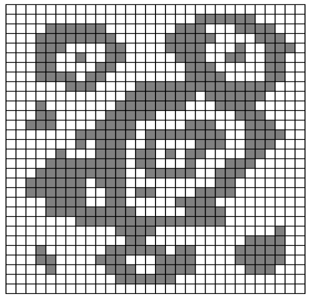
1: $\theta(\log n * \log n)$, 2: $\theta(\log \log n)$, 3: $\theta(n^{1/2} \log \log n)$, 4: $\theta(n \log \log n)$, 5: $\theta(n^2 \log \log n)$,
6: Ingen av alternativene 1-5.

Oppgave 2.

Denne oppgaven vil teste dine evner til å lese og forstå et gitt Javaprogram der rekursjon anvendes. Spørsmålene er knyttet til hva programmet gjør på et konkret datasett. Du vil her videre få muligheter til å anvende dine teoretiske kunnskaper på en praktisk problemstilling:

Tenk deg et satellitt-bilde der vi ser et utsnitt av et kystlandskap. Svart rute betyr "land", hvit rute betyr "vann". Vi skal anta at hele randen på bildet er **havet** (vann). Vi ønsker å finne en rekke egenskaper ved slike bilder, og har laget et Java-program for dette:

i\j 0 1 .... N (x-retningen (j))



(y-retningen(i)) N Øyriket er omgitt av havet.

Figur 1. **(boolean**[][] map)

```
class Islands
{
    public static final int MAX_SEGMENT = 1000;
    public static final int UNINITIALIZED = -1;
    public static final int PROBLEMSIZE = 30;

    public static int discoverSegment( boolean[][] map, int[][] numberMap,
                                       boolean type, int id, int posX, int posY )
    {
       if( posY >= 0 && posY < PROBLEMSIZE &&
           posX >= 0 && posX < PROBLEMSIZE &&
           numberMap[ posY ][ posX ] == UNINITIALIZED &&
           type == map[ posY ][ posX ] ){

          numberMap[ posY ][ posX ] = id;

          return 1 +
             discoverSegment( map, numberMap, type, id, posX + 1, posY ) +
             discoverSegment( map, numberMap, type, id, posX - 1, posY ) +
             discoverSegment( map, numberMap, type, id, posX, posY + 1 ) +
             discoverSegment( map, numberMap, type, id, posX, posY - 1 );

       } else {
          return 0;
       }
    }

    public static void main( String[] args )
    {
       int[] pi = new int[ MAX_SEGMENT ];
       int[] a = new int[ MAX_SEGMENT ];
       int n = 0;

       boolean[][] map = /* 30 x 30 array from Figure 1. */;
       int[][] numberMap = new int[ PROBLEMSIZE ][ PROBLEMSIZE ];

       for( int i = 0; i < PROBLEMSIZE; i++ )
          java.util.Arrays.fill( numberMap[ i ], UNINITIALIZED );

       for( int i = 0; i < PROBLEMSIZE; i++ ){
          int parent = -1;
          for( int j = 0; j < PROBLEMSIZE; j++ ){
             if( numberMap[ i ][ j ] == UNINITIALIZED ){
                pi[ n ] = parent;
                a[ n ] = discoverSegment( map, numberMap,
                                          map[ i ][ j ], n, j, i );
                n += 1;
             }
             parent = numberMap[ i ][ j ];
          }
       }
    }
}
```

Figur 2 (Programmet Islands)

Bildet i Figur 1 har 5 øyer som grenser til havet, totalt 5 innlandsøyer og totalt 6 innsjøer. Den største innsjøen ligger på den største øya og har 2 innlandsøyer. En av disse har en egen innsjø med en innlandsøy i.

a) På øya øverst til venstre i Figur 1 finner du et vann bestående av 13 ruter. På vannet er det en øy som er 1 rute stor. Vis i hvilken rekkefølge programmet behandler de 13 vann-rutene ved å skrive tallene 1,2,...,13 i rutene på anvist figur i besvarelsen.

b) Hva er verdiene i variablene a[5] og a[14] når programmet terminerer ?

c) Vis verdiene Pi[8],Pi[9],Pi[10] i tabellen `int[] pi = new int[ MAX_SEGMENT ];)` etter at programmet har terminert.

d) Pi-tabellen er en vektor/array og representerer en velkjent datastruktur. Hva er den presise benevnelsen for denne?

e) Hvordan ville du generelt gå fram for å løse oppgaven: *"Finn det maksimale antallet øyer i en av innlandssjøene."* (Du skal her kun skrive en meget kort og meget presis forklarende tekst.)

f) Anta at et bilde består av N x N ruter. Hva er da tidskompleksiteten til programmet vårt (Figur 2), uttrykt ved O-notasjonen? Du skal her besvare oppgaven ved å gi et lavest mulig siffer (1,2,...,8) blant følgende 8 alternativer: (Entydig svar.)

   1:O(N),2:O(NlogN),3:O(N^2),4:O(N^2logN),5:O(N^3),6:O(N^4),7:O(2^N),8:O(4^N)

g) Enkelte typer kyst-bilder vil vise seg å by på problemer for programmet vårt. Gi et minimalt lite eksempel på et slikt bilde. Forklaring er unødvendig.

<u>Oppgave 3.</u>

Studenten Gro tar 3 fag, der sannsynligheten for å stryke i alle fag, ifølge egen vurdering (Tabell 1), er $0.8 \times 0.75 \times 0.9 = 0.54$, altså 54%, <u>uten</u> ekstra studietimer pr. uke. Gro synes ikke dette er helt bra og setter opp en tabell som viser hvordan ekstra leseinnsats kan redusere sjansene for å stryke i de enkelte fagene. (Ikke lur på hvordan hun fant fram til tallene.)
Vi tar det her for gitt at sannsynligheten for stryk avtar monotont med økende innsats i form av flere studietimer, men innser at dette ikke nødvendigvis gjelder i det virkelige liv.

Nb.: Det kreves her ingen kunnskaper i statistikk ut over det at sannsynligheten for at flere begivenheter skal inntreffe samtidig er lik produktet av de enkelte sannsynlighetene, se f.eks. regnestykket ovenfor.

Studietimer pr uke	Sannsynligheten for stryk (blir % ved å mulitiplisere med 100)		
	Fransk (Fag nr 1)	DigDat (Fag nr 2)	AlgDat (Fag nr 3)
0	0.8	0.75	0.9
1	0.7	0.7	0.7
2	0.65	0.67	0.6
3	0.62	0.65	0.55
4	0.6	0.62	0.50

Tabell 1

Målet til Gro er å minimalisere sannsynligheten for at hun stryker i alle de 3 fagene. Hun har 4 ekstra studietimer pr uke å fordele på fagene, og må da regne ut hvordan dette bør gjøres for å oppnå målet sitt. Gro blir glad da hun oppdager at kanskje *Dynamisk Programmering* fra AlgDat kan hjelpe henne. Gro starter med å nummererer fagene 1,2,3, som vist i Tabell 1.

Tabell 1 (der t=0,1,2,3,4 og f=1,2,3) består da av 5× 3=15 tall, der

p(t,f) = ”sannsynligheten for å stryke i fag nr. **f** dersom **t** timer/uke brukes på dette ene faget”

Gro innfører videre betegnelsen

Pmin(t,f) = ”minimal sannsynlighet for å stryke i fag nr. **f** og fag med høyere numre dersom **t** timer/uke er til disposisjon”

Med disse betegnelsene er det greitt å innse at Pmin(t,3) = p(t,3)

a) Hvilken verdi får Pmin(0,2)?

b) Hvilken verdi får Pmin(1,2)?

c) Hva blir den rekursive formelen for beregning av Pmin(t,f) ?

d) Hvor mange av de inntil 4 ekstra Studietimene lønte det seg for Gro å sette av til hvert av de 3 fagene? (Ikke nødvendigvis avhengig av svaret i c).

e) Hva blir for Gro den minimale sannsynligheten for å stryke i alle de 3 fagene ? (Altså stryk i samtlige 3 fag.)