Vastaa tentin järjestäjän antamalle paperille (ei kysymyspaperille). Kirjoja, laskinta tms. ei saa olla tentissä. Kukin tehtävä on 1 tai 2 pisteen arvoinen. Muiden kuin koodaa tai piirrä -tehtävien mallivastaukset ovat melko lyhyet, 1, . . . , 4 riviä.

Jos pisteytysohje tuottaa negatiivisen tuloksen, annetaan 0 pistettä. Jos pisteytysohje tuottaa yli maksimi pistettä, annetaan maksimin verran.

1. Taulukko A indeksoidaan $0,\ldots,n-1$. Kirjoita pseudokoodi tai ohjelmanpätkä, joka kääntää A:n sisällön takaperin. Esimerkiksi merkkejä sisältävästä taulukosta $[\texttt{k},\texttt{u},\texttt{r},\texttt{s},\texttt{s},\texttt{i}]$ pitää tulla $[\texttt{i},\texttt{s},\texttt{s},\texttt{r},\texttt{u},\texttt{k}]$.

   ```
   for( i = 0; i < n/2; ++i ){
     alkio apu = A[i]; A[i] = A[n-1-i]; A[n-1-i] = apu;
   }
   ```

   Pisteytys: Ensimmäinen virhetoiminto $-\frac{1}{2}$ (poikkeus: vain tyhjällä taulukolla ilmenevä $-\frac{1}{4}$), seuraavat $-\frac{1}{4}$. Kuitenkin kirjoitusvirhe josta on helppo arvata mitä tarkoitettiin (esim. `j++` kun piti olla `j--`) $-\frac{1}{4}$. Turha tehottomuus (aputaulukon käyttö) ±0. A:n nimen vaihto ±0. Turha monimutkaisuus ±0.

2. Puolitushausta on olemassa eri versioita. Miksi joitakin niistä ei voida testata siten, että etsitään avaimen paikka taulukosta jollain muulla, luotettavalla tavalla, ja verrataan testikohteen vastausta siihen?

   Avain saattaa olla taulukossa useasti, ja testikohde ja luotettava etsintä saattavat löytää eri kohdat.

3. Onko INSERTIONSORT:in suoritusaika $O(n^3)$? Onko se $\Theta(n^3)$?

   Se on $O(n^3)$. Se ei ole $\Theta(n^3)$.
   Pisteytys: Oikea $+\frac{1}{2}$, väärä ±0. Esim. ”on $O(n^2)$” ±0, koska se ei vastaa kysymykseen. Selvästi väärä ylimääräinen väite $-\frac{1}{4}$.

4&5. Alla on esitetty valintajärjestäminen C++-koodina. Olkoon $n =$ `A.size()`. Mitä i:n arvosta tiedetään rivin 3 alussa? Mitä i:n, j:n ja p:n arvoista tiedetään rivin 5 alussa? Mitä i:n ja p:n arvoista tiedetään rivin 7 alussa?

Rivin 3 alussa pätee $0 \le i < n-1$.
Rivin 5 alussa pätee $0 \le i \le p < j < n$.
Rivin 7 alussa pätee $0 \le i \le p < n$ ja $i < n-1$.
Pisteytys: Mahdollisimman tiukka oikea $+\frac{1}{4}$, muu tosi ± 0, väärä $-\frac{1}{4}$. Kuitenkin $0 \le i$ tuottaa pisteitä enintään kerran. "$p = \ldots$" tms. tulkitaan yhdeksi vääräksi.

```
1  void Valintajarjesta( taulukko & A ){
2    for( unsigned i = 0; i+1 < A.size(); ++i ){
3      unsigned p = i;
4      for( unsigned j = i+1; j < A.size(); ++j ){
5        if( A[j].x < A[p].x ){ p = j; }
6      }
7      alkio apu = A[i]; A[i] = A[p]; A[p] = apu;
8    }
9  }
```

6&7. Anna valintajärjestämisen ulommalle silmukalle invariantti!

Koko taulukossa on alkuperäiset alkiot. Osa $A[0 \ldots i-1]$ on kasvavassa järjestyksessä. Joko $i = 0$ tai jokainen osan $A[i \ldots n-1]$ alkio on vähintään yhtäsuuri kuin $A[i-1]$. Pätee $0 \le i < n$.
Pisteytys: Alkuperäisyysehto $+\frac{1}{2}$, sama osataulukolle ± 0. Kasvavuusehto $+1$, myös koskien $A[0 \ldots i]$. Osien välinen ehto $+1$, mutta jos $A[i]$ on mukana alkuosassa niin vain $+\frac{1}{2}$, koska voi olla $A[i] > A[i+1]$. Väite $0 \le i < n$ tuottaa $+\frac{1}{2}$ mutta $0 \le i < n-1$ vain $+\frac{1}{4}$, koska silmukan loppuessa $i = n-1$.

8. Anna valintajärjestämisen sisemmälle silmukalle invariantti! (Ei tarvitse toistaa mitään siitä, mitä ulomman silmukan invariantti sanoo.)

Jokainen osan $A[i \ldots j-1]$ alkio on vähintään yhtäsuuri kuin $A[p]$.
Pisteytys: Kukin virhe alueen päissä $-\frac{1}{4}$.

9. Onko valintajärjestäminen vakaa? Anna esimerkki syötteestä, joka osoittaa, että se ei ole vakaa, tai perustele, että sellaista syötettä ei voi olla olemassa!

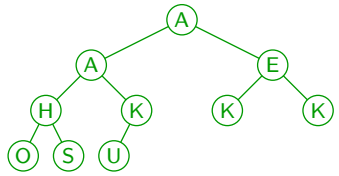
Ei ole. Vastaesimerkki: $[2, 2, 1]$.

10. Kuinka paljon valintajärjestäminen käyttää aikaa nopeimmillaan ja hitaimmillaan Θ-merkinnällä ilmaistuna? Perustele lyhyesti.

    Sekä hitaimman että nopeimman tapauksen ajan kulutus on $\Theta(n^2)$. Kun $n > 1$, rivi 4 suoritetaan $n + (n-1) + \ldots + 1 = \frac{1}{2}n^2 + \frac{1}{2}n$ kertaa, eikä mitään riviä suoriteta useammin.
    Pisteytys: Hitaimmillaan $\frac{1}{4}$, nopeimmillaan $\frac{1}{4}$, sisemmän silmukan kierrosmäärän vaihtelu hallittu perustelussa $\frac{1}{2}$.

11&12. Piirrä keko, jossa on alkiot H, A, U, S, K, A, K, E, K ja O. Käytä aakkosjärjestystä pienin ensimmäisenä eli ylinnä. Piirrä se sekä taulukkomuodossa että puumuodossa.

A	A	E	H	K	K	K	O	S	U



13. Jos kasvavan taulukon alkioille varattu muistialue loppuu kesken, niin ohjelman suoritusympäristö kasvattaa sitä automaattisesti. Miksi se ei kasvata sitä joka kerta aina samalla määrällä (esim. 100 alkiolla)?

    Ison taulukon täyttämisestä tulisi hidasta. Se veisi aikaa $\Theta(n^2)$, koska joka kasvatuksessa kopioidaan koko vanha sisältö.
    Pisteytys: Jos hitaus on mainittu mutta ei mitenkään viitattu että se on pahempi kuin vakiokerroin, niin $\frac{3}{4}$. Viittaus muistin tuhlaamiseen ± 0, koska kysymys sallii kasvatusmäärän olla pieni.

Käännä

14. Minkä verran kasvavan taulukon muistialue tyypillisesti kasvaa kerrallaan?

    Uusia alkioita on tyypillisesti sama määrä kuin alkioita oli entuudestaan.
    Pisteytys: Vaikka ei ole totta että määrä on aina kahden potenssi, sai siitä silti täyden pisteen, koska tätä yksityiskohtaa ei ole kurssilla kerrottu. (Käyttäjä voi asettaa minkä tahansa koon, josta automatiikka aloittaa kaksinkertaistamisen.)

15&16. Pikajärjestämisen (Quicksort) käyttämä ositus jakaa taulukon kahteen tai kolmeen osaan. Mitä ominaisuuksia osituksen lopputuloksella täytyy olla?

Kukin alkuosan alkio on enintään yhtäsuuri kuin mikään muiden osien alkio. Kukin keskiosan alkio on enintään yhtäsuuri kuin mikään loppuosan alkio. Yhteensä osat sisältävät alkuperäiset alkiot, eikä muuta. Ainakin kaksi osaa on epätyhjiä.
Pisteytys: Suuruusjärjestysehto +1. Epätyhjyysehto +1. Alkuperäisyysehto $+\frac{1}{2}$.

17. Miten Dijkstran algoritmi ylläpitää reitin etsinnän aikana tietoa, jonka avulla reitti voidaan lopuksi tulostaa?

Jokaisessa solmussa (paitsi lähtösolmussa) on linkki reitin edelliseen solmuun.

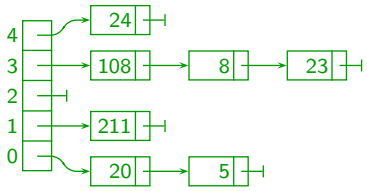
18. Miten A* reitinetsintäalgoritmi eroaa Dijkstran algoritmista?

A* käyttää apuna arviota jäljellä olevasta matkasta maaliin.
Pisteytys: Viittaus heuristiikkaan kertomatta, että kyse on loppumatkan (tai kokonaismatkan) arviosta $\frac{3}{4}$.

19. Mikä on laskentajärjestämisen (counting sort) pahin heikkous?

Se edellyttää, että avaimet ovat pieniä kokonaislukuja.
Pisteytys: Viittaus isoon muistin käyttöön ilman viittausta avainten arvoalueeseen $\frac{1}{2}$.

20. Piirrä hajautustaulu, jonka hajautusfunktiona on x mod 5 eli `x % 5`, ja jossa on alkiot 211, 23, 5, 20, 24, 8 ja 108.



21&22. Osoitin `os` osoittaa binäärihakupuun solmuun. Kirjoita pseudokoodi tai ohjelmanpätkä, joka muuttaa `os`:n osoittamaan edelliseen solmuun.

```
if( os->vasen ){
  os = os->vasen;
  while( os->oikea ){ os = os->oikea; }
}else{
  while( os->vanh && os == os->vanh->vasen ){ os = os->vanh; }
  os = os->vanh;
}
```

Pisteytys: Kumpikin haara 1. Vastaus aivan muuhun puusta etsintätehtävään $\frac{1}{2}$ jos se toimii ja $\frac{1}{4}$ jos se on sinnepäin.

23. Kerro yksi hajautustaulujen etu verrattuna punamustiin puihin.

Alkion löytää avaimen perusteella keskimäärin ajassa $O(1)$.

24. Kerro yksi punamustien puiden etu verrattuna hajautustauluihin.

Alkioita voi selata avainten mukaisessa suuruusjärjestyksessä.

25. Tarkastellaan mitä tahansa polkua punamustan puun juuresta osoittimeen, joka osoittaa ei minnekään. Miksi siinä on vähintään sama määrä mustia kuin punaisia solmuja?

Juuri on musta. Jos punaisella solmulla on lapsia, ne ovat mustia. Siksi korkeintaan joka toinen solmu on punainen.
Pisteytys: Punaisten sääntö $\frac{3}{4}$, juuri on musta $\frac{1}{4}$. Punaisten säännön tilalla oleva ”joka toinen on musta ja joka toinen punainen” $\frac{1}{2}$ ja lisänä oleva $-\frac{1}{4}$. Kukin muu epätosi väite $-\frac{1}{4}$.

26&27. Syötteessä on n alkiota, joista pitää löytää k pienintä. Luku k on suuri ja n on sitäkin paljon suurempi. Kuvaile nopea algoritmi, ja kerro sen ajan kulutus. (Ei tarvitse kirjoittaa pseudokoodia tms.)

Tähän on useita mahdollisuuksia:

A. Nopean mediaanin etsinnän periaatteella eli vain yhdellä osataulukolla jatkavalla Quicksortilla järjestetään taulukkoa niin kauan, että k:s alkio on löytynyt. Tällöin sitä pienemmät (ja mahdollisesti yhtäsuuria) ovat sen vasemmalla puolen. Keskimäärin $\Theta(n)$, hitaimmillaan $\Theta(n^2)$.

B. Tehdään taulukosta keko pienin ylinnä ajassa $\Theta(n)$. Otetaan k alkiota pois ajassa $O(k \log n)$. Ajan kulutus $O(n + k \log n)$.

C. Ylläpidetään k pienintä alkiota keossa suurin ylinnä. Ajan kulutus $O(n \log k)$.

D. Järjestetään taulukko hyvällä algoritmilla ja poimitaan k ensimmäistä. Ajan kulutus $O(n \log n)$.

E. Käydään aineisto läpi ja ylläpidetään k pienintä järjestetyssä taulukossa. Ajan kulutus $O(nk)$.

Pisteytys: Algoritmeista A ... C enintään $+\frac{3}{2}$, D enintään $+1$, E enintään $+\frac{1}{2}$. Ajan kulutus enintään $+\frac{1}{2}$. Asiavirheistä miinusta.

Seuraavien kysymysten tavoitteena on kurssin kehittäminen. Niihin ei ole ennalta määrättyä oikeaa vastausta, vaan mikä tahansa asiallinen vastaus ja sen perustelu tuovat pisteen.

28. Binääripuun korkeutta on luentoruuduissa merkitty kirjaimella k, koska sana ”korkeus” alkaa sillä. Kirjallisuudessa käytetään usein kirjainta h, koska korkeus on englanniksi ”height”. Pitäisikö mielestäsi luentoruuduissa käyttää k vai h? Mistä syystä k on sinusta parempi / huonompi?

    Molemmille oli kannatusta, mutta h voitti selvästi. Englanti on alan valtakieli.

29. Mitä asiaa olisit halunnut kurssiin lisää, ja miksi?

    Vastaukset hajaantuivat laajalti, eikä mitään asiaa mainittu kovin monta kertaa. Joidenkin mielestä toistettiin turhaan Algoritmit 1:n asioita, mutta toisista vastauksista (ja mm. tämän tentin tuloksista) syntyy vaikutelma, että niitä olisi pitänyt kerrata enemmänkin. Luentosisältö ei tukenut riittävän hyvin ohjelmointiharjoitusten tekemistä varsinkaan dynaamisen ohjelmoinnin ja välitulosten muistamisen osalta.

    Odotukset kurssin tavoitteista näyttivät moninaisilta. Osa olisi halunnut paljon pieniä ohjelmointitehtäviä yksittäisten asioiden oppimiseksi, ja toisessa ääripäässä olisi haluttu oppia enemmän algoritmien valitsemista ja soveltamista. (Kirjassa oli toistakymmentä pientä ohjelmointitehtävää mallivastauksineen, mutta ehkä olisi haluttu automaattitarkastettavia tehtäviä.) Kurssin ohjelmointitehtäväthän ovat nykyisellään välimuoto: kokonaisuutena isohkoja ja soveltavia, mutta sisältävät runsaasti yksityiskohtien kanssa näpräämistäkin.

30. Minkä asian voisi jättää kurssilta pois, jotta saataisiin tilaa asialle, jota haluaisit lisää?

    Paljon ääniä saivat ei mitään, invariantit sekä koodin tai algoritmin tai sen toiminnan yksityiskohdat. Toisaalta joissakin vastauksissa todettiin, että yksityiskohdat voi opiskella itsekin. Moni aihe sai hajaääniä.

loppu