Linköpings Tekniska Högskola
Institutionen för Datavetenskap
Filip Strömbäck

XX XX 2019

TDDI16
Datastrukturer och algoritmer
Exempeltentamen (DAT1)
2019-XX-XX, 08–12

Examinator:	Erik Nilsson
Jour:	Filip Strömbäck.
Antal uppgifter:	9
Max poäng:	40 poäng
Preliminära gränser:	Betyg 5 = 35p, 4 = 27p, 3 = 20p.
Hjälpmedel:	Inga hjälpmedel tillåtna!

VÄNLIGEN IAKTTAG FÖLJANDE

- Observera att betygsgränserna kan komma att justeras, i samtliga kurser.
- Du får själv välja om du vill skriva din lösning på papper eller på dator.
- Lösningar till olika problem skall placeras enkelsidigt på separata blad, eller i egen fil. Skriv inte två lösningar på samma papper eller i samma fil. Delproblem får dela papper / fil.
- Om inte annat framgår ska indexering av arrayer / listor börja från 0.
- Papper: Sortera lösningarna innan de lämnas in.
- Filer: Skicka in som lösning till rätt problem i tentaklienten, och döp filen till ett passande namn (exempelvis uppg1.txt).
- **MOTIVERA DINA SVAR ORDENTLIGT:** avsaknad av, eller otillräckliga, förklaringar resulterar i poängavdrag. **Även felaktiga svar kan ge poäng** om de är korrekt motiverade.
- Om ett problem medger flera olika lösningar, t.ex. algoritmer med olika tidskomplexitet, ger endast optimala lösningar maximalt antal poäng.
- Papper: Lämna plats för kommentarer.
- SE TILL ATT DINA LÖSNINGAR/SVAR ÄR LÄSLIGA.

Lycka till!

1. **Prioritet och köer** (6 p)

Onlinecasinot AllYourMoneyAreBelongTo.us har öppnat portarna. Efter en mycket lyckad lansering har de fått massor av kunder, vilket gjort att spelen ibland laggar och det kan vara svårt att komma in. Istället för att expandera och åtgärda problemen har de kommit på att de kan starta en premiumserver som maximalt tillåter 1000 spelare åt gången. För att komma in måste man ställa sig i kö, och det kostar några extra kronor att bli insläppt. Man kan dessutom köpa ett VIP-medlemsskap som gör att man får gå före vanliga medlemmar i kön. Därutöver ger man anställda företräde, till och med över VIP-medlemmar.

Under designprocesen stötte de på problem. De vill göra premiumservern så effektiv och snabb som möjligt, och eftersom det visade sig att deras designers inte var speciellt bra på just den biten har de valt att kalla in extern hjälp. Enligt ryktet är du en framgångsrik DALG-student, och på dessa grunder har de valt just dig. Mot skälig ersättning går du med på att ta jobbet.

Det första problemet du ställs inför på ditt nya jobb är att designa kösystemet. Du kan utgå från att det finns existerande datatyper för spelare.

(a) Vilken (eller vilka) datatyp(er) väljer du att använda för att designa kön och varför? Motivera! (3)

(b) Vilken tidskomplexitet har din kö (insättning och borttagning)? (1)

(c) Efter en tid visade det sig att ingen ville betala de hutlösa summorna man krävde för att köpa VIP-medlemsskap och casinot har beslutat sig för att skrota hela idén. Därtill har inte de anställda skött sig speciellt väl (utom du, självfallet, men du skulle aldrig få för dig att spela casinospel) och man har beslutat sig för att ta bort deras fördelar i kösystemet. Alla ska alltså, från och med nu, köa på samma villkor till premiumservern. Vilka förändringar gör du i representationen av kösystemet? Motivera! (2)

2. CD-samlingen (6 p)

Du har en stor samling CD-skivor i din bokhylla hemma. För att hålla ordning på skivorna, och för att enkelt kunna hitta rätt skiva har du införskaffat en hylla med fack för de individuella skivorna, liknande den som visas i bilden till höger.

För att enkelt hitta bland alla skivor har du sorterat dem alfabetiskt efter artist och sedan efter albumets titel. För att underlätta hanteringen har du numrerat alla fack. Facket längst upp till vänster har nummer 0, facket nedanför nummer 1, och så vidare. Skivorna är också märkta med en siffra som beskriver i vilket fack skivan hör hemma, så att du snabbt vet var skivan ska vara även om inte alla skivor är i hyllan för närvarande.

Överlag har systemet fungerat mycket bra sedan du införskaffade hyllan och märkte upp alla skivor, och du är mycket nöjd över din briljanta idé. Den enda nackdelen med systemet är att det är en relativt stor arbetsinsats att introducera nya skivor i ditt system. Tyvärr råkade du nyligen hitta en stor samling skivor på en loppmarknad i grannskapet som du inte kunde undvika frestelsen att köpa. Det du inte tänkte på just då var den markanta arbetsinsatsen som kommer krävas för att sortera in och märka upp alla nya skivor...

Din hylla för CD-skivor med en del av din samling (Creative Commons) `https://commons.wikimedia.org/wiki/File:CD_cases_on_rack.jpg`

Efter en lång sömnlös natt av funderande inser du att arbetet skulle förenklas mycket om du hade ett program där du helt enkelt kunde mata in alla CD-skivor och sedan kunde programmet få berätta för dig i vilket fack varje skiva ska vara. Då slipper du åtminstone att sortera alla skivor manuellt! Du tänker dig följande arbetsgång:

1: Mata in namnet på alla CD-skivor i programmet medan du plockar fram dem och lägger dem i en stor hög på golvet.

2: Informera programmet om att du har matat in alla skivor.

3: Plocka upp en skiva från högen, fråga programmet vilket nummer skivan ska ha, märk upp den och stoppa in den i hyllan. Upprepa tills alla skivor har blivit märkta.

Det enda som återstår nu är att skriva programmet!

(a) Beskriv hur du kan implementera programmet som kan hjälpa dig med sorteringen enligt ovan. Tänk på att du inte vill behöva vänta på programmet i steg 3, så programmet måste kunna svara på den frågan snabbt! Beskriv och motivera dina val av datastrukturer och algoritmer. (2)

(b) Vilken tidskomplexitet har din lösning, uttryckt i det totala antalet CD-skivor (N), för att svara på frågan i steg 3? (2)

(c) Vilken tidskomplexitet har din lösning, uttryckt i det totala antalet CD-skivor (N), för hela processen? Alltså: vad är den totala tidskomplexiteten för inmatning av N stycken skivor följt av N stycken frågor inklusive eventuellt arbete som programmet behöver göra i steg 1 och 2? (2)

3. **Sorteringsalgoritmer** (2 p)

Svaren behöver ej motiveras.

Efter **ett fåtal** iterationer av några olika sorteringsalgoritmer på den osorterade arrayen *Original* i tabellen nedan (iteration i bemärkelse fullständig körning av inre loop, alternativt rekursivt anrop) har vi resultaten i *1, 2, 3* och *4*.

Original:	30	92	2	55	5	91	41	99	26	68	22	8	0	72	25
1:	2	30	55	92	5	91	41	99	26	68	22	8	0	72	25
2:	72	68	41	55	25	8	0	30	26	5	22	2	91	92	99
3:	0	2	5	55	92	91	41	99	26	68	22	8	30	72	25
4:	0	2	5	22	8	25	41	72	26	68	55	30	91	99	92
Sorterad:	0	2	5	8	22	25	26	30	41	55	68	72	91	92	99

Matcha delvis sorterad array mot en algoritm. Felaktig matchning ger minuspoäng, dock kan uppgiften ej ge total minuspoäng. För full poäng krävs 4 korrekta svar.

(a) Insertionsort

(b) Heapsort

(c) Selectionsort

(d) Quicksort, elementet längst till höger i partitionen används som pivot

4. **Para ihop strumpor** (2 p)

Du har just tvättat alla dina strumpor. Som vanligt har några försvunnit i tvätten, så du står inför det vanliga problemet att para ihop de strumporna som inte har kommit bort. Tidigare har du helt enkelt parat ihop strumporna på måfå, men det har lett till dömande blickar från dina modemedvetna kompisar. Därför vill du den här gången även ta hänsyn till strumpornas färg!

Beskriv hur du effektivt kan komma fram till hur många par strumpor du har kvar efter tvätten. Varje par måste vara i samma färg. Beskriv vilken datarepresentation du använder, men välj själv en lämplig representation. Notera även vad din lösning har för tidskomplexitet, där n är antalet strumpor.

5. **Den rättvisa festen** (4 p)

Du arrangerar en fest för dina kompisar. Det finns många olika typer av drycker: Coca-cola, kaffe, te... vatten... etc. Framåt kvällen framkommer det att folk har druckit olika mycket av olika saker, vilket självfallet orsakar missnöje bland gästerna! Som snäll värd vill du så klart lösa problemet på ett smidigt sätt.

För att lösa problemet har du baserat på dina inventarier räknat ut hur mycket av varje dryck var och en av gästerna kan få så att alla kan få lika mycket. Dessutom har du sammanställt en lista av vad var och en av gästerna har druckit. Dessvärre är listorna i oordning eftersom konsumptionen har skett i olika ordning (och minnet sviktar, inte alla hanterar koffein så bra...).

Det enda som är kvar för att alla ska bli nöjda är att dela ut de drycker som är kvar för att alla ska uppnå sin rättvisekvota. Du har därför bett gästerna att ställa sig i kö för att de ska få sin beskärda del av drycken. För att inte skapa ytterligare missnöje vill du hitta ett sätt att snabbt lista ut vilka drycker du ska ge till personen som står längst fram i kön för att denne ska bli nöjd. Antag att du har två listor tillgängliga för att lösa problemet, dels listan av vilka drycker varje gäst borde ha fått samt listan av vad den nuvarande gästen har konsumerat. Exempelvis:

Rättvisekvota:	kaffe	te	kaffe	te	Coca-cola	Pepsi-cola	vatten	vatten
Konsumerat:	Coca-cola	kaffe	vatten	te	kaffe			

I detta fallet borde gästen få: te, vatten och Pepsi-cola.

Notera: Du kan inte anta att varken rättvisekvotan eller listan över konsumerade drycker är sorterade. Du har också fått i dig en rejäl dos koffein...

Notera: Du kan anta att ingen gäst har fått mer än sin rättvisekvota.

(a) Beskriv din lösning. Motivera dina (ditt) val! (3)

(b) Vilken tidskomplexitet har din lösning? Alltså, tidskomplexiteten för dig att komma fram till vad en specifik gäst ska få (inte samtliga). Antag att n är det maximala antalet drycker som någon har druckit. (1)

6. **Hashtabeller** (4 p)

Vi har en hashtabell med linjär adressering och några element instoppade. Hashfunktionen är h(x) = x mod size (arrayindex står under arrayen för tydlighets skull).

0	20	null	13	null	25	6	7	18	26
0	1	2	3	4	5	6	7	8	9

(a) I vilken ordning har elementen stoppats in? Det finns flera lösningar. Ge fyra korrekta lösningar för maxpoäng. (2)

(b) Om vi tar bort 18 från den delvis fyllda hashtabellen ovan (remove / delete), hur kommer tabellen se ut? Det finns flera olika lösningar. Ge två olika lösningar som inte hashar om hela tabellen utom möjligen i värsta fallet, och förklara dem, för maxpoäng. (2)

7. **Algoritmer och Tidskomplexitet** (6 p)

(a) Beräkna tidskomplexiteten med avseende på n för följande funktion: (1)

```
int fun(n) {
     int res = 0;
     for (int i = 1; i < 2*n; ++i) {
          res++;
     }
     return res;
}
```

(b) Beräkna tidskomplexiteten med avseende på n för följande funktion: (1)

```
int more_fun(n) {
     int res = 0;
     for (int i = n; i >= 1; i=i/2) {
          res += i;
     }
     return res;
}
```

(c) Beräkna tidskomplexiteten med avseende på n för följande funktion: (2)

```
int super_fun(n) {
     int res = 0;
     for (int i = 3; i < n; ++i) {
          res += fun(2*i) + more_fun(i);
     }
     return res;
}
```

(d) Du har skrivit ett program som ritar ut kvadrater i terminalen m.h.a ASCII-tecken. Programmet frågar först efter en storlek och skriver sedan ut kvadraten med storlek 1, 2 osv. till och med storleken som matades in. Beskriv med hjälp av ett uttryck i Ordo-notation hur många tecken som skrivs ut om talet n matas in. Du behöver inte ta hänsyn till nyradstecken eller inmatning. (2)

Exempel med inmatning av talet 3:

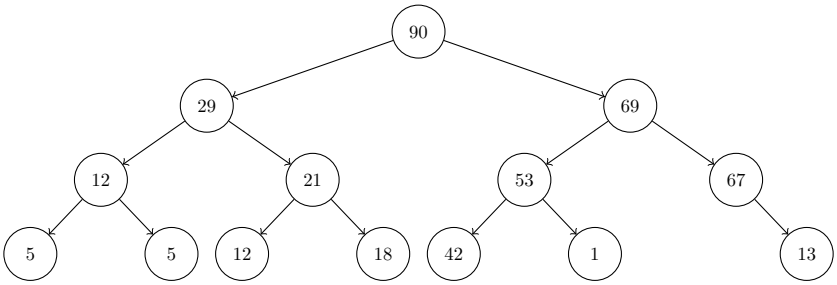
```
+--+
|  |
+--+
+----+
|    |
|    |
+----+
+------+
|      |
|      |
|      |
+------+
```

Tips: Om du tycker att det känns konstigt att räkna tecken, tänk dig att utskrift av ett tecken tar konstant tid och räkna sedan den totala tiden som utskriften tar. Det är ekvivalent med antalet tecken som skrivs ut.

Tips: Skriv kod som löser problemet och analysera den!

8. **Träd** (6 p)

Studera följande träd:



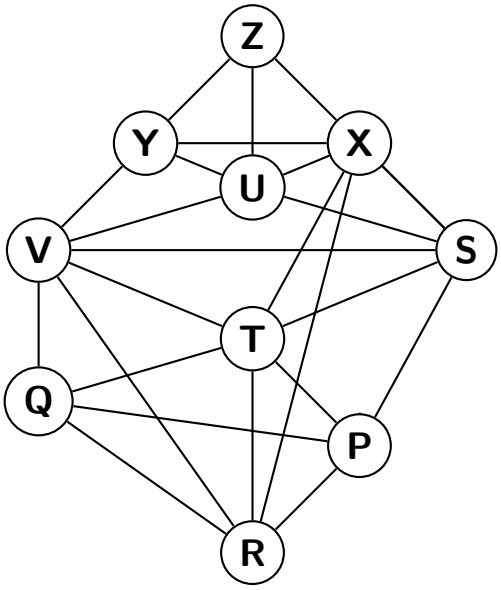
(a) Är trädet ett binärt sökträd? Motivera! (2)

(b) Är trädet balanserat? Motivera! (2)

(c) Är trädet en min-heap, max-heap eller ingetdera? Motivera! (2)

9. **Graftraversering** (4 p)

Studera följande graf:



Det räcker med en korrekt lösning på a) och b). Du behöver alltså inte räkna upp flera tänkbara lösningar.

(a) I vilken ordning kommer noderna att traverseras med bredden först sökning, om vi utgår från R? (2)

(b) I vilken ordning kommer noderna att traverseras med djupet först sökning, om vi utgår från R? (2)