KTH CSC, Linda Kann

DD1320/1325 TENTAMEN I TILLÄMPAD DATALOGI

Torsdagen den 7 januari 2016 kl 14–18

Hjälpmedel: En algoritmbok och ditt eget formelblad. För betyg E krävs att alla E-uppgifter är godkända, för betyg C krävs utöver detta betyg C på alla C-uppgifter och för betyg A krävs högsta betyg på alla uppgifter. Lycka till!

1. *Otursautomat*

Betyg E. Det vore ju oturligt om ditt företagsnamn redan var upptaget. Konstruera en KMP-automat som söker efter `TURIOTUREN`

Rita upp automaten och ange next-vektorn.

(10 min)

2. *Tidskomplexitet*

Betyg E. Para ihop algoritmerna nedan med rätt tidskomplexitet.

algoritm	***komplexitet***
Skriva ut alla möjliga binära tal som är n bitar långa.	$O(logn)$
Sortera n element med mergesort.	$O(n)$
Pusha n element på en stack.	$O(nlogn)$
Sortera in ett nytt element i ett binärt sökträd med n element.	$O(2\sup n)$

a.

b.

c.

d.

(10 min)

3. *Binärt sökträd*

Betyg E. Rita upp ett binärt träd som uppfyller följande fem villkor: Trädet ska innehålla åtta olika värden. Trädet ska vara ett binärt sökträd. Trädet ska vara balanserat. Trädets höjd ska vara tre. Trädet ska ha fyra löv.

(10 min)

4. *Vilken datastruktur?*

Betyg E. Att använda rätt datastruktur till algoritmen handlar mer om kunskap än tur. Skriv upp tre olika korrekta meningar givet alternativen nedan:
Vid *xxx* är det lämpligt att använda en *yyy*.

xxx	***yyy***
breddenförstsökning	stack
djupetförstsökning	prioritetskö
bästaförstsökning	kö

(10 min)

5. *Oturssyntax*

Betyg E. Givet syntaxen nedan:

```
<skrock> ::= <pred> " " <obj> " "  | <pred> " " <obj> " " <indobj>
<pred> ::= "lägga" | "se" | "spilla" | "krossa" | "gå under"
<obj> ::= "nycklar" | "en svart katt" | "salt" | "en spegel" | "stegen"
<indobj> ::= "på bordet" | "på vägen" | "på golvet"
```

Avgör vilken/vilka av följande meningar som accepteras:

```
se en svart katt
gå under på golvet
lägga salt på vägen
```

(10 min)

6. *Kryptering*

Betyg E. One-time pad är en krypteringsmetod som fungerar så här:
Givet ett meddelande på binär form

- Slumpa fram en nyckel med lika många binära siffror som meddelandet har
- Gör bitvis *xor* mellan meddelandet och nyckeln

Att ta *xor* med samma nyckel på det krypterade meddelandet avkodar.

Avkoda följande krypterade meddelande (uppdelat i 8 bitsstycken för läslighetens skull)

```
meddelande = 10111101 10111110 10111011
slumpad nyckel = 11111111 11111111 11111111
```

Översätt sedan meddelandet till text enligt följande A har nr 65 vilket blir 01000001 binärt.
B har nr 66 vilket blir 01000010 osv.

(10 min)

7. *Hashning*

Betyg C. Vi har en hashtabell med 10 platser, och hashfunktionen
$h(s) = 2 * len(s)\%10$

Demonstrera inhashning av följade fem fembokstaviga ord
prova
kliva
ramla
gråta
snyta
med linjär respektive kvadratisk probning. Till din hjälp i den kvadratiska probningen kan du använda följande tabell (du behöver inte proba längre än den räcker).

k	k^2	$\sum_{i=1}^{k} i^2$
1	1	1
2	4	5
3	9	14
4	16	30
5	25	55

Diskutera fördelar och nackdelar med respektive *krockhanteringsmetod* i det här fallet. (Hashfunktionen förbättrar vi en annan gång.)

(20 min)

8. *Jämför sortering*

Betyg C. Insättningssortering, bubbelsortering, quicksort och räknesortering har du sett användas för sortering av en array (Python-lista).

Tänk dej nu att du istället har värdena i en abstrakt kö med operationerna enqueue, dequeue och isEmpty.

Vilka av dessa sorteringsmetoder skulle då fungera? Rita och berätta för var och en av sorteringsmetoderna hur sorteringen av kön skulle gå till alternativt vad det är som inte fungerar. Analysera också hur tidsåtgången för sorteringsalgoritmen ändras.

(20 min)

9. *Kortstack*

I kortspel som bluffstopp representeras kortleken bäst med en stack.

Den abstrakta stacken `stack` har anropen `push()`, `pop()` och `isEmpty()`.

Vi vill skriva en funktion `antal` sådan att `antal(stack)` returnerar antalet element i stacken. `antal` ska fungera oberoende av hur stacken är implementerad.

a. Formulera rekursiv algoritm som omedelbart kan omsättas i fungerande programkod! Efter anropet ska stacken ha samma innehåll som före anropet.

b. Visa med ett konkret exempel hur din rekursiva tanke fungerar

c. Fungerar din algoritm lika bra för en abstrakt kö? Förklara!

10. *Reguljära uttryck*

Betyg A. Ett reguljärt uttryck kan innhålla tecken inom klamrar, t ex [b-d], för att ange att något av de angivna tecknen ska matchas.

Exempel: sökströngen [b-d][aeiou][s] matchar orden bas, bus, cis, dis, dos och dus

Formulera en algoritm som hittar alla matchande strängar i en ordlista. Beskriv de datstrukturer du använder, och uppskatta tidskomplexiteten.

Indata: en sökträng med m stycken klammeruttryck (det är tre i exemplet ovan) en ordlista med n stycken m-bokstavsord (där n är mycket större än m)

Utdata: Alla ord i ordlistan som matchar sökströngen

(30 min)