Hemtentamen i Objektorienterad Programmering, DAT043

Joachim von Hacht

Datum: 2020-06-10

Tid: 08.30-12.30

Hjälpmedel: Allt

Betygsgränser:

U: -23, 3:24-37, 4:38-47, 5:48-60 (max 60)

Lärare: Joachim von Hacht, tel. 031/772 10 03. Alla frågor under tentamen skickas med epost till hajo@chalmers.se

Granskning: Meddelas via Canvas.

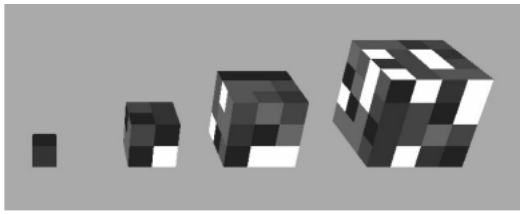
Instruktioner:

- SKRIV ALLA SVAR I KODEN I RESPEKTIVE FIL: Q1andQ2, Q3, Q4, ... direkt i IntelliJ-projektet. Om du inte har tillgång till IntelliJ skriv lösningar i vilken typ av program som helst men lägg in lösningarna i IntelliJ-projektet. Filerna skall heta som ovan (Q1and2, Q3, ..)
- Lösningarna måste klara de fall som anges *samt fall som är principiellt lika*. Lösningar som bara klarar exemplen räcker *inte*. Överkomplicerade lösningar kan ge poängavdrag.
- Färdiga klasser m.m. som får användas anges för varje uppgift. Anges inget får man alltid använda de grundläggande språkliga konstruktionerna, arrayer, egna metoder och egna klasser (men inte t.ex. List/ArrayList/String/Math/m.fl.).

LYCKA TILL...

1. Uppgift 1 och 2 är sammanslagna till en enda uppgift (denna).

   Antag att vi klistrar ihop små kuber med sidlängd 1 till större kompakta kuber. Hur många små kuber går det åt om vi vill ha en kub av varje sort med sidlängd från 1 till n? Figuren nedan visar de fyra kuberna i det första exemplet nedan ($n = 4$). Tillsammans innehåller de $1 + 8 + 27 + 64 = 100$ småkuber. 4p

   Skriv ett program för detta problem. Programmet ska fråga efter talet n, där $2 \leq n \leq 40$, och skriver ut antalet små kuber som går åt. Exempel

   ```
   Talet n > 4
   Antal: 100
   Talet n > 7
   Antal: 784.
   ```

2. —

3. Skriv en metod som givet en icke tom array med heltal returnerar en array med alla "dominanta" element i den givna arrayen. Ett dominant element definieras som ett element där alla element till höger om är mindre än det dominanta. Elementet längst till höger räknas alltid som en dominant. För full poäng krävs en lösning *utan* nästlade loopar. Exempel: 6p

Orginal-array	Resultat-array (dominanta)
[14, 9, 11, 7, 8, 5, 3]	[14, 11, 8, 5, 3]
[1, 2, 3, 4, 5]	[5]
[5, 4, 3, 2, 1]	[5, 4, 3, 2, 1]
[5, -4, 3, 2, 1]	[5, 3, 2, 1]
[1, 1, 1, 1]	[1]
[1]	[1]

4. Skriv en metod som givet en kvadratisk heltalsmatris och ett heltal, n, avgör om det finns någon kvadratisk delmatris vars summa av element blir n. Hela matrisen räknas som delmatris till sig själv (oäkta delmatris). För full poäng krävs funktionell nedbrytning. 10p

```
int[][] m = {     (0 och 1 bara för enkelhets skull
   {0, 1, 0, 1},   kan vara andra elementvärden)
   {0, 0, 1, 1},
   {1, 0, 1, 1},
   {0, 1, 0, 0}
};
```

```
Anrop                        Resultat
-----                        -------
hasSubmatrixWithSum(m, 0)    true   (1x1 matris)
hasSubmatrixWithSum(m, 4)    true   (2x2)
hasSubmatrixWithSum(m, 5)    true   (3x3)
hasSubmatrixWithSum(m, 7)    false
hasSubmatrixWithSum(m, 8)    true   (4x4)
hasSubmatrixWithSum(m, 9)    false
```

5. Ibland vill man jämföra hur ljudmässigt lika två namn är (låter). Detta kan åstadkommas genom att koda namnen på ett visst sätt. Namn med samma eller närliggande kod "låter" lika. T.ex. får "robert" och "rupert" samma kod (vi antar engelska namn med enbart små bokstäver). Följande metod kan användas: 12p

   - Spara undan den första bokstaven.
   - Ersätt alla tecken a, e, i, o, u, y, h, w i namnet med tecknet 0 (inte värdet, vi jobbar hela tiden med tecken och strängar) och alla konsonanter enligt tabellen nedan (inklusive första bokstaven).
   - Ersätt alla följder av siffrertecken med en tecken. T.ex. "222" blir "2".
   - Ta bort alla 0-tecken.
   - Om den sparade (första) bokstaven har samma kod som det första tecknet i resultatet, ta bort det första tecknet i resultatet. Om resultatet nu är kortare än 3 tecken lägg till 3 stycken 0-tecken sist.
   - Slå ihop det sparade första tecknet med de tre första siffrorna i resultatet. Detta är koden, se exempel nedan.

   Konsonanter:

Bokstav	Kodtecken
b, f, p, v	1
c, g, j, k, q, s, x, z	2
d, t	3
l	4
m, n	5
r	6

   Exempel på kodningar av namn.

```
Namn          Kod
----          ----
Rupert        R163
Robert        R163
Tymczak       T522
Zakk          Z200
Honeyman      H555
Bo            B000
```

   Implementera en metod som givet ett namn (String) returnerar koden (String) för detta. Alla metoder i Appendix är tillåtna.

   TIPS: Skapa en hjälpmetod som givet ett tecken ger koden (i from av ett tecken). Låt metoden använda två strängar enligt:

```
String keys   = "abcdefghijklmnopqrstuvwxyz";
String values = "01230120022455012623010202"; // Jämför tabell ovan
```

6. Rita en bild som visar variabler, värden, referenser och objekt samt hur dessa förhåller sig till varann före, respektive efter anropet av metoden doIt. Rita som vi ritat under kursen; namn, lådor, pilar o.s.v. Strängobjekt kan ritas förenklat t.ex. "abc". Rita bilden i något ritprogram eller för hand och lägg in den i IntelliJ-projektet med namn: Uppg6.png (eller gif eller jpg). Kontrollera att bilden är skarp och läsbar. 8p

```
A a1 = new A("a", null);
A a2 = new A(a1.str + "b", a1);  // Before
doIt(a2);                // Call
                           // After

void doIt(A a) {
   A tmp = new A(a.str + "c", a.a);
   a.a = tmp;
}

class A {
   String str;
   A a;
   A(String str, A a) {
      this.str = str;
      this.a = a;
   }
}
```

7. Skapa en objektorienterad modell av en uthyrningsfirma för släpvagnar (där man t.ex. kan hyra vagnar att koppla efter bilen om man skall flytta). Klasserna skall vara så icke-muterbara som möjligt och dölja så mycket som möjligt av sin data (information hiding). Alla klasser/metoder i Appendix är tillåtna. Följande klasser skall användas;

```
public enum Status { FREE, OCCUPIED } // Status for Trailer

public class Customer {    // Someone renting a Trailer
   private String id;
   private String name;
    // Constructor, setter/getter, equals, hashCode omitted
}

public class Period {     // The renting period
   private Date start;
   private Date end;
   // Constructor, setter/getter, equals, hashCode omitted
}

public class Trailer {    // The trailer to rent
   private Status status;
   public Status getStatus() { return status; }
   public void setStatus(Status status) { this.status = status;}
   // Constructor, setter/getter, equals, hashCode omitted
}
```

a) Skapa en klass Location, som representerar en plats där man kan hämta eller lämna släpvagnar. En Location har en adress och ett antal släpvagnar. Lägg till en metod som returnerar en ledig släpvagn om en sådan finns. Annars null. Inga setter/getter, equals eller hashCode behöver anges, vi antar att dessa finns. Gäller även nedan. 3p

b) Skapa en klass för en bokning av en vagn. En bokning består av en vagn, en kund, en plats att hämta och en att lämna, samt en bokningsperiod. All data sätts då bokningen skapas. 2p

c) Skapa en klass TrailerRental för firman. Firman har ett antal hämtnings/lämningsställen och ett antal bokningar. Lägg till en metod som givet hämtnings och lämningsställe, en kund och en hyrperiod skapar (och sparar) en bokning om det är möjligt. D.v.s. platserna finns och det finns en ledig vagn att hämta. Om det lyckas returneras sant (och vagnen markeras som upptagen). Annars returneras falskt. 5p

8. Givet en heltals-array skriv en metod som returnerar längden på den längsta delsekvensen med enbart ökande värden. Exempel (sekvensen/längden behöver inte vara unik): 10p

Array	Ökande delsekvens	Max längd
1	1	1
1, 2	1, 2	2
5, 4, 3, 2, 1	t.ex 3	1
2, 4, 3, 7, 4, 5	2, 3, 4, 5	4
10, 9, 2, 5, 3, 7, 6	t.ex. 2, 3, 7	3
4, 5, 1, 7	4, 5, 7	3

TIPS:

- Använd en extra array, len, som håller reda på max sekvenslängd till det aktuellt element i arrayen. Initiera alla element i len till 1 (minsta möjliga längd för alla delsekvenser).
- Utgå från följande resonemang.
  - Antag att vi bara tittar på de två första elementen i och j. Om arr[i] är större än arr[j] så finns det en ökande delsekvens med längd 2 till arr[i]. Uppdatera len[i] till 2.

    ```
    arr: 2,4,3,7,4,5    len: 1,1,1,1,1,1 (start)
         j i                 1,2,1,1,1,1 (uppdaterad)
    ```

  - Antag att vi fortsätter att jämföra element parvis på samma sätt, om arr[i] är större än arr[j] så finns en delsekvens på längd len[j] + 1 till arr[i], eftersom len[j] var max längd till j. D.v.s. till arr[i] (7) nedan finns en sekvens 2 + 1 = 3. Uppdatera len[i] till 3. Dock får vi se upp, det kan finnas en annan sekvens till arr[i] som ev. är längre. Om så uppdaterar vi inte len[i].

    ```
    arr: 2,4,3,7,4,5
           j   i             1,2,2,3,1,1
    ```

  - Längen av längsta sekvensen ges nu av maxvärdet i len.

Tillåtet att använda Math.max() och Arrays.fill()

APPENDIX

```
Färdiga Java-klasser/metoder som får användas om så anges vid uppgiften.

Ur klassen String
- equals(s), avgör om en sträng innehåller samma tecken som en annan.
- charAt(int i), ger tecknet vid index i.
- indexOf(char ch), ger index för tecknet ch, -1 om tecknet saknas.
- length() ger längden av strängen.
- subString(int start, int end), ger en delsträng från
  start (inkl.) till end-1.
- subString(int start), ger en delsträng från start (inkl.)
  till strängens slut.
- toCharArray(), gör om strängen till en array med tecken
Ur klassen StringBuilder
- append(String s), lägger till strängen s sist i
  Stringbuilder-objektet.
- append( char ch ), som ovan
- charAt(), samma som String
- toString(), omvandlar StringBuilder-objektet till en String.
Ur List/ArrayList
- get(i), ger objektet för index i
- add(o), lägger till objektet o sist i listan
- set(i, o), lägger till objektet vid index i (skriver över).
- remove(o), tar bort objektet o ur listan, returnerar
  true om detta lyckades annars false
- remove(i), tar bort och returnerar objektet vid index i ur listan
- removeAll( list ), tar bort alla element i list.
- contains(o), sant om objektet o finns i listan.
- indexOf(o), ger index för objektet
- size(), ger längden på listan
Ur Math
- sqrt(n), roten ur n.
- max(n1, n2) ger det största av värdena n1 och n2
Ur Character
- isDigit(ch), isLetter(ch), isWhiteSpace(ch), getNumericValue(ch),
  toString(ch)

Klassen Random med metoden nextInt() är alltid tillåten.
```