Hemtentamen i Objektorienterad Programmering DAT043

Joachim von Hacht

Datum: 2020-08-19

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).

   Skriv ett program som frågar efter de tre inledande talen i en talföljd och därefter skriver ut de tio första talen. De tre första talen är positiva heltal mindre än 100. För talföljden gäller att förändringen av skillnaden mellan talen i följden är konstant. Exempel: 5p

```
Talföljd          Skillnader      Förändring skillnad
--------          --------        -------------------
1,3,6,10,15,...   2,3,4,5,...      1
1,2,3,4,5,...     1,1,1,1,...      0
10,7,2,-5,-14,..  3,5,7,9,...      2
```

   Exempel programkörningar:

```
Första talet ? 1
Andra talet ? 5
Tredje talet ? 11
Talföljden: 1, 5, 11, 19, 29, 41, 55, 71, 89, 109,

Första talet ? 10
Andra talet ? 7
Tredje talet ? 2
Talföljden: 10, 7, 2, -5, -14, -25, -38, -53, -70, -89,
```

2. —

3. Givet två heltals-arrayer skriv en metod som returnerar de element som finns i båda arrayerna. Inga eventuellt duplicerade element skall tas med. Exempel: 7p

```
a1 = [1, 2]
a2 = [3, 4]
a3 = [1, 2, 3, 4]
a4 = [1, 2, 2, 3, 6, 6, 6]
a5 = [7, 5, 3, 6, 2, 2, 3, 6]

Anrop             Resultat
-----             --------
inBoth(a1,a2)     []
inBoth(a1,a3)     [1, 2]
inBoth(a2,a4)     [3]
inBoth(a4,a5)     [2, 3, 6]
```

4. Skriv en metod som givet en kvadratisk heltalsmatris och ett heltal returnerar sidlängden på den minsta kvadratiska delmatris där summan av elemenen är lika med det givna talet. Saknas sådan returneras -1. För full poäng krävs funktionell nedbrytning. Exempel: 12p

```
int[][] m = {{0, 1, 0, 3},
             {2, 0, 2, 1},
             {1, 0, 3, 1},
             {0, 2, 0, 2}};

Anrop                      Resultat
-----                      -------
smallestSubMatrix(m, 3)     1
smallestSubMatrix(m, 6)     2
smallestSubMatrix(m, 11)    3
smallestSubMatrix(m, 8)    -1
```

5. Vi vill undersöka om en robot, efter en given promenad, når en viss position. Följande gäller: 8p

   - Roboten rör sig i ett 2D-koordinatsystem (roboten har en x och y position).
   - Riktning anges m.h.a. tecknen: 'n', 'e', 's', 'w' för respektive nord, öst, syd och väst.
   - Robotens promenad ges av en sträng med ett antal av delsträckor. Varje sträcka inleds med en bokstav och därefter ett antal siffror. Bokstaven anger riktning enligt ovan. Siffrorna beskriver ett positivt heltal som anger antal steg roboten skall gå i given riktning.

   Skriv en funktion, boolean willHit(int startX, int startY, endX, endY, String walk), som givet robotens startposition (startX och startY) samt vägen (walk) svarar sant om roboten stannar på den givna slutposition (endX och endY) efter att ha vandrat promenaden. Om ej returneras falskt. Alla metoder i Appendix är tillåtna. Exempel:

```
Anrop                                   Resultat
----                                    -------
willHit(0, 0, 0, 0, "e1w1s1n1")         true
willHit(1, 1, 1, 1, "e12w12s12n12")     true
willHit(0, 0, 0, 0, "e123w1")           false
willHit(0, 0, 0, 0, "e122w123e1")       true
willHit(2, 2, 8, 3, "n3e4s2e2")         true
```

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(new int[]{1, 2});
A a2;            // Before
a2 = doIt(a1);   // Call
                  // After
A doIt(A a) {
   A tmp = new A();
   a.i = tmp.i;
   return tmp;
}

class A {
   int[] i;
   A() { i = new int[]{0, 0};}
   A(int[] i) { this.i = i;}
}
```

7. Skapa en objektorienterad modell av spelet Scrabble (Alfapet). För detaljer om spelet se t.ex. https://en.wikipedia.org/wiki/Scrabble. Alla klasser skall vara så icke-muterbara som möjligt och dölja all data. Alla klasser/metoder i Appendix är tillåtna.

Följande klasser skall användas:

```
class Tile {      // A tile to place on the board
   char letter;   // The letter used to form a word
   int points;    // The points for the letter
   // Constructor, set/get, equals/hashCode omitted
}
class Board {         // The board to put tiles on
   boolean put(Tile tile, int row, int col) {
    // Will put tile at row, col. If success return true else false
   }
}
```

a) Skapa en klass Bag för påsen som innehåller alla brickor (Tile) som ingen spelare tagit än. Alla brickor "läggas i" påsen då den skapas. Lägg till metoderna isEmpty, ger sant om påsen är tom, och getTile som returnerar en slumpvis bricka. Inga setter/getter, equals eller hashCode behöver anges, vi antar att dessa finns. Gäller även nedan. 2p

b) Skapa en klass Rack (hållare) för de brickor en spelare har. En hållare kan ha max 7 brickor. Rack skall ha en metod add som lägger till en bricka, om detta lyckas returneras true annars false. Lägg också till och en metod remove, som tar bort en angiven bricka och returnerar den (om möjligt, annars null). 3p

c) Skapa en klass för en spelare. En spelare har ett namn och en hållare. Namn sätts då spelaren skapas. 1p

d) Skapa en klass Scrabble som representerar hela spelet. Spelet har ett antal spelare, en spelplan (Board), en påse och en aktuell spelare. Lägg till en metod addToRack som låter aktuell spelare plocka en bricka ur påsen och placera den på sin hållare. Lägg dessutom till en metod putOnBoard som tar en bricka från aktuell spelares hållare och placerar den på spelplanen. I båda fallen returneras sant om det lyckades annars falskt. 6p

8. Besvara frågorna nedan.

a) Betrakta klasserna X och Y nedan. Vad skrivs ut om följande fyra rader exekveras? Förklara ytterst noggrant varför utskriften blir som den blir. 4p

```
X x = new Y();  // What will be printed? Why???
x.doIt(5);
Y y = new Y();
y.doIt(5.0);

class X {
   void doIt(double d) { out.println("doIt A " + d); }
}

class Y extends X {
   void doIt(int i) { out.println("doIt B " + i);}
}
```

b) Betrakta klasserna A, B och C nedan. Vad skrivs ut om följande kodrad exekveras. Förklara ytterst noggrant varför utskriften blir som den blir. 4p

```
C c = new C(1);  // What will be printed why???

class A {
   int i;
   A(int i) { this.i = i; out.println("ctor A");}
}

B extends A {
   static C c = new C(4);
   B(int i) { super(i); out.println("ctor B"); }
}

C extends B {
   C(int i) { super(i); out.println("ctor C");
   }
}
```

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)
Ur Integer
- parseInt(str), omvandlar strängen str till en int.
Klassen Random med metoden nextInt() är alltid tillåten.
```