Tentamen i Objektorienterad Programmering, DAT043

Joachim von Hacht

Datum: 2019-08-21

Tid: 08.30-12.30

Hjälpmedel: Engelskt-Valfritt språk lexikon

Betygsgränser:

U: -23

G: 24-43

VG: 44-60 (max 60)

Lärare: Joachim von Hacht, tel. 031/772 10 03. Någon besöker ca 09.30 och 11.00

Granskning: Tentamen kan granskas på studieexpeditionen. Vid ev. åsikter om rättningen eposta mig och ange noggrant vad du anser är fel så återkommer jag (ev. ta en bild och skicka med).

Instruktioner:

- För full poäng på essäfrågor krävs ett läsbart, begripligt och heltäckande svar. Generellt 1p för varje relevant aspekt av problemet. Oprecisa eller alltför generella (vaga) svar ger inga poäng. Konkretisera och/eller ge exempel.
- Det räcker med enbart relevanta kodavsnitt, övrig kod ersätts med "..." (aldrig import, main-metod, etc....). Vi utgår från att användaren alltid skriver rätt och/eller gör rätt (d.v.s ingen felhantering behövs). Om felhantering skall ingå anges detta specifikt.
- 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.

LYCKA TILL...

1. Förklara med en eller ett par meningar. Du får gärna ge kodexempel eller rita. 4p

   a) Referenslikhet

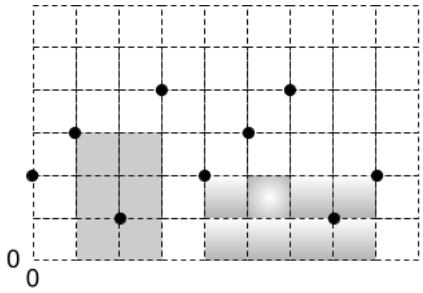
   b) this

2. Ett av kodavsnitten a) eller b) ger kompileringsfel. Vilket ger fel? Varför? 2p

```
a)
public int triple(int i){ return 3 * i; }
public static int inc(int i){ return triple(i) + 1; }

b)
public static int triple(int i) { return 3 * i;}
public int inc(int i) { return triple(i) + 1;}
```

3. En array kan användas för att ange punkter i ett koordinatsystem. Index anger x-värde och elementvärde anger y-värde. T.ex ger arrayen [2, 3, 1, 4, 2, 3, 4, 1, 2] punkterna i bilden nedan. 8p



Utifrån två punkter i arrayen och x-axeln kan man skapa en rektangel genom att dra en horisontell linje utifrån det minsta y-värdet och därefter beräkna arean (gråmarkerade i bilden). Bilden visar;

Rektangel	Punkt1	Punkt2	Area
Vänster	1,3	3,4	2*3 = 6
Höger	4,2	8,2	4*2 = 8

Implementera en metod som givet en array enligt ovan returnerar den största rektangel som kan skapas (max area). Du får använda metoderna Math.max(a,b) och Math.min(a,b) vilka ger det största respektive minst värdet av a och b.

4. Givet en kvadratisk matris m med icke-negativa heltal och sidan, $s > 1$ samt ett heltal $n > 1$. Skriv en metod som avgör om det finns någon kvadratisk delmatrix med sidan n i m sådan att alla värden i delmatrisen är stigande då man genomlöper den. För full poäng krävs funktionell nedbrytning. Exempel: 12p

```
Matris          n   Returvärden
[1, 2, 4]       2   true  (har flera delmatriser med sidan 2)
[6, 7, 9]       3   true  (har en delmatris med sidan 3)
[10, 12, 13]

[1, 3, 2, 1]    2   true  (har flera delmatriser med sidan 2)
[4, 5, 1, 7]    3   false (har ingen delmatris med sidan 3)
[1, 2, 6, 0]    4   false (har ingen delmatris med sidan 4)
[1, 7, 9, 2]
```

5. En parentesgrupp är en sträng med matchande, ev. nästlade parenteser som inleds med en vänsterparentes t.ex (), (()) eller ((((())))). Givet en icke-tom sträng med parentesgrupper, skriv en metod som skalar av de yttersta parenteserna för varje grupp i strängen. Exempel: 8p

```
Inträng                 Utsträng
"()"                     ""
"(())"                  "()"
"(())(())"              "()()"
"(())((()))()()(())"    "()(())()"
```

Alla metoder i Appendix är tillåtna.

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”. 8p

```
A[] as = new A[2];
as[0] = new A(1);
as[0].n = new A(3);  // Before
doIt(as[0]);    // Call
                // After

void doIt(A a) {
    a.n = new A(2);
    a.n.p = a;
}

class A {
    int i;
    A p;
    A n;
    A(int i) { this.i = i;}
}
```

7. Vi skall skapa en objektorienterad modell för en kurs. Följande klasser skall användas; 10p

```
public enum Role {STUDENT, TEACHER, TA};  // TA = teaching assistant

public class Result {     // Used by LADOK class below
    private String studentId;
    private String courseId;
}

public class LADOK {      // Class to store results
    public static LADOK getInstance() {
       // Will return the one and only instance of this class
    }
    public Result getResultFor(Person p, String courseId) {
       // Will return result for p for courseId if found else null
    }
}
```

a) Skapa en klass för en person. En person har ett id och en roll (Role). Båda sätts då personobjektet skapas. Vi kan anta att equals, hashCode och alla nödvändiga set/get-metoder finns (du behöver inte skriva dessa, gäller även nedan).

b) Skapa en klass för en kurs. En kurs har ett id, ett antal studenter, ett max antal studenter som kan antas och en referens till LADOK (m.m. vi utelämnar en hel del ...). Id, LADOK och max antal studenter anges då kursen skapas. Lägg till en metod register som låter en person registrera sig för att delta i kursen. Bara studenter får registrera sig (en gång) och alla måste få plats. En student med tidigare resultat på kursen får inte registrera sig. Om registreringen lyckas så returnerar metoden true annars false.

c) Visa hur du i ett program kan skapa en ny kurs och hur man registrerar en student för kursen.

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.

8. Betrakta koden nedan och ange för varje rad a) - g) *en* av följande. 8p

   - Kompilerar ej
   - Körningsfel
   - Om inget av ovan, ange vad som skrivs ut.

   Du måste ge en kort motivering till varje resultat (flera satser på varje rad p.g.a. utrymme)!

```
a) IX ix = new A(); ix.doIt(1.0);
b) C c = (C) new D(); c.doIt(1);
c) IY iy = new D(); iy.doIt(1.0);
d) A[] arr = new A[3]; arr[0] = new B(); arr[0].doIt(1);
e) B b = new C(); IY iy = (IY) b; iy.doIt(1);
f) A a = new D(); a.doIt(1.0);
g) IX ix = new D(); B b = (B) ix; b.doIt(1);  (2p)

interface IX { void doIt(double d);}
interface IY { void doIt(int i);}

class A {
   public void doIt(double d) { out.println("doIt A"); }
}
class B extends A implements IX {
   public void doIt(int i) { out.println("doIt B");}
}
class C extends B {
   public void doIt(int i) { out.println("doIt C");}
}
class D extends B implements IY {
   public void doIt(double d) { out.println("doIt D"); }
}
```