Ordinarie tentamen för ID1004 Objektorienterad programmering, 28 oktober 2014, kl 9-13

Denna tentamen examinerar 3.5 högskolepoäng av kursen.

Inga hjälpmedel är tillåtna.

Tentamen består av en obligatorisk del och en extra del. För ett godkänt betyg på hela tentamen måste den obligatoriska delen vara godkänd. Detta berättigar till betyg E på tentamen. För högre betyg kan ytterligare uppgifter lösas i den extra delen. Examinator förbehåller sig möjligheten att justera tentamensbetyget med hänsyn till en bedömning av helhetsintrycket.

Den obligatoriska delen ger maximalt 47 poäng. För godkänt krävs 30 poäng eller mer.

Läs noga igenom alla frågorna först och planera tiden.

Om inget annat uttryckligen sägs så visas och efterfrågas kod och uttryck ur Java version 6 eller 7. Var noga med metoders returtyper.

Obligatorisk del

Uppgift 1 (1+2+1 poäng)

a) Deklarera en statisk konstant som heter ANTAL_KANALER och initiera den till sexton. Välj en lämplig datatyp och sätt åtkomstskydd så att den är allmänt tillgänglig.

b) Deklarera och initiera två instansvariabler som kan lagra heltal. Den första variabeln ska vara initierad till halva värdet av konstanten ANTAL_KANALER. Den andra variabeln ska vara initierad till värdet av den första variabeln plus ett. Dessa två variabler ska bara användas inuti sin definitionsklass. Deklarera en lämplig åtkomstbegränsning.

c) Deklarera en array av flyttal. Den ska ha ett element för varje kanal (enligt konstanten ovan), när första kanalens nummer är 1.

Uppgift 2 (2+3 poäng)

Följande variabler finns:

```
String tarEn = ”tar en”;
String allan = ”Allan”;
String kaka = ”kaka”;
```

a) Deklarera en ny strängvariabel och initiera den med ett stränguttryck i vilket de givna variablerna ingår, så att texten i den nya strängen blir *Allan tar en kaka*.

b) Använd en `StringBuilder` (se bilaga) och dess metod `append` för att konstruera strängen som initierar variabeln. Texten ska ha samma lydelse som i uppgift (a).

Uppgift 3 (3 poäng)

Skriv en metod `minsta` som tar emot en icke tom array med double, och returnerar index för det minsta värdet (eller ett av de minsta om det råkar finnas flera) i arrayen. Längden på en array `a` erhålls med `a.length`;

Uppgift 4 (3 poäng)

Skriv en metod `minstaUrKortaste` som tar två icke tomma arrayer med double, och returnerar det minsta värdet ur den kortaste arrayen. Metoden `minsta` från uppgift 3 ska anropas som en del av lösningen.

Uppgift 5 (3 poäng)

Givet följande kod:

```
int in = /* godtyckligt värde 0-9999 */
int a = in;
int b = a * 3;
int c = b + 1;
int d = c * 3;
int e = d + in;
int f = e / 10;
```

Vilket värde får variabeln f?

Tips: påbörja analysen vid tilldelningen av f och substituera mot början.

Uppgift 6 (4 poäng)

Skriv färdigt metoden

```
public static boolean equalsIgnoreCase (String s1, String s2)
```

Metoden returnerar `true` om de två strängarna har samma lydelse, oavsett om de enskilda tecknen har stor eller liten bokstav. Metoderna `int String.charAt(int i)` och `char Character.toUppercase(char c)` ska användas. En strängs längd erhålls med metoden `int String.length()`.

Uppgift 7 (2 poäng)

Fyll i tabellens tredje kolumn med uttryckets sanningsvärde (true eller false) (svara på separat papper).

a	b	(a | b) & !(a & b)
false	false	
true	false	
false	true	
true	true	

Uppgift 8 (2+2+2 poäng)

Givet dessa metoder:

```
public static void foo (int a) {
  System.out.print ("U");
}
public static void foo (String s) {
  System.out.print ("L");
}
public static void foo (int a, String s) {
  System.out.print ("B");
}
public static void foo (char c) {
  System.out.print ("E");
}
```

a) Vad kommer att skrivas ut när dessa metodanrop exekveras?

```
foo(0, "WHO");
foo(420);
foo("DR");
foo("-3");
foo('#');
```

b) Skriv ytterligare en metod `foo` som på samma sätt genererar bokstaven A.

c) Skriv en sekvens med anrop till metod `foo` som genererar utskriften LABB.

Uppgift 9 (15+2 p)

Klassen `Point` representerar en punkt i planet. Den ser ut så här:

```
class Point {
  protected double x, y;

  public Point (double x, double y) {
    this.x = x;
    this.y = y;
  }

  public double getX () {
    return x;
  }

  public double getY () {
    return y;
  }

  public double distance (Point p) {
    double xd = this.x - p.x;
    double yd = this.y - p.y;
    return Math.sqrt (xd * xd + yd * yd);
  }

  public String toString () {
    return "(" + x + ", " + y + ")";
  }
}
```

a) Är det nödvändigt att använda `this.x` i konstruktorn? Motivera svaret.

b) Är det nödvändigt att använda `this.x` i metod `distance`? Motivera svaret.

Klassen `Point` ligger till grund för två ytterligare klasser, `NamedPoint` och `Vector`. Klassen `NamedPoint` har följande utseende:

```
class NamedPoint extends Point {
  private String name = "";

  public NamedPoint (String name, double x, double y) {
    super (x, y);
    this.name = name;
  }

  public String getName () {
    return name;
  }

  public String toString () {
    return name + ":" + super.toString ();
  }
}
```

c) Vilka metoder ärver klass `NamedPoint` från klass `Point`?

d) Vilka metoder åsidosätts av klass `NamedPoint`?

e) Vilka metoder och/eller konstruktorer i klass `Point` anropas från klass `NamedPoint`?

f) Vilka metoder är unika för klass `NamedPoint`?

Klass `Vector` ser ut på detta sätt:

```
class Vector extends Point {

  private static final Point origin = new Point (0, 0);

  public Vector (double x, double y) {
    super (x, y);
  }

  public Vector add (Vector v) {
    return new Vector (this.x + v.x, this.y + v.y);
  }

  public Vector mul (double a) {
    return new Vector (a * x, a * y);
  }

  public double length () {
    return distance (origin);
  }

  public String toString () {
    return "V" + super.toString ();
  }
}
```

g) Vilka metoder åsidosätts av klass `Vector`?

h) Vilka metoder och/eller konstruktorer i klass `Point` anropas från klass `Vector`?

i) Vilka metoder är unika för klass `Vector`?

j) Vilken utmatning ger följande kod?

```
Point p0 = new Point (4, 5);
Point p1 = new Point (1, 1);
System.out.println (p0);
System.out.println (p1);
System.out.println (p0.distance (p1));
```

k) Vilken utmatning ger denna kod?

```
Point p0 = new NamedPoint ("A", 4, 5);
Point p1 = new Point (1, 1);
System.out.println (p0);
System.out.println (p1);
System.out.println (p0.distance (p1));
```

l) Vilken utmatning ger denna kod?

```
Vector v0 = new Vector (4, 5);
Vector v1 = new Vector (1, 1);
System.out.println (v0);
System.out.println (v1);
System.out.println (v0.add (v1));
```

m) Vad skrivs ut av den här koden?

```
Point [] pa = new Point [3];
pa[0] = new Point (3, 3);
pa[1] = new NamedPoint ("B", 2, 6);
pa[2] = new Vector (4, 5);
for (Point p : pa)
  System.out.println (p);
```

n) Arrayen `pa` innehåller objekt av flera olika typer. Förklara hur det är möjligt?

o) Skulle det fungera att lägga till denna kodrad till koden i uppgift m? Motivera svaret.

```
double len = pa[2].length();
```

p) Antag att vi vill kunna ha både namngivna punkter och vektorer, och att namngivna klasser ska kunna hanteras uniformt med avseende på deras namn (och endast namnet). Föreslå en strategi för att uppnå det.