Tentamen

DD2385 Programutvecklingsteknik vt 2009

Tisdagen den 26 maj 2009 kl 10.00 – 13.00
Inga hjälpmedel utom penna, sudd och linjal

Tentans del I omfattar 22 poäng. Del II har 23 poäng

Preliminära betygsgränser (gränserna blir inte högre):

Betyg FX:	15-16 poäng på del I
Betyg E:	17-22 poäng på del I
Betyg C:	17-22 poäng på del I och 11-23 poäng på del II

Betyg FX får kompletteras till C om C-nivån på del II är nådd på tentan.

del I

1. (5p) Rita ett UML-klassdiagram som åskådliggör följande klasser. Alla relationer ska vara med. För full poång måste öppna och slutna pilspetsar samt streckade och heldragna linjer användas rätt. Instansvariabler och metoder behöver *inte* vara med i diagrammet men ev. relationer som ges av variabler och metoder ska ritas ut. Alla klasser och interface utom utom String måste vara med i diagrammet. De långa klassnamnen får förkortas.

```
interface Spelmodell {
    void drag(int i, int j);
    String[][] ställning();
}

class Luffarschacksmodell implements Spelmodell { ... }

class Femtonspelsmodell implements Spelmodell { ... }

class Spel extends JFrame implements ActionListener {
    int n;
    JButton[][] spelplan; // kvadratisk spelplan n*n
    Spelmodell spelmod;

    Spel (Spelmodell spmod, int n) {} // konstruktor

    public void actionPerformed (ActionEvent e) {...}
}
```

2. (1p) Klassen `Spel` i föregående fråga implementerar ett spel på en kvadratisk rutindelad spelplan. Spelets regler finns i ett särskilt objekt i `Spel` som refereras med `spelmod`. Om man med hjälp av klasserna ovan skriver ett program som kan växla mellan olika spelmodeller i samma körning av programmet, vilket designmönster använder man då ?

3. (2p) Stegtävlingar som motionsaktivitet är populära just nu. Anställda på en arbetsplats delas in i lag, får var sin stegräknare och går och går och går så mycket de orkar och rapporterar varje kväll in dagens antal steg till en server skriven i Java. Ställningen mellan lagen redovisas kontinuerligt på en webbsida. Det lag som först har gått t.ex. från Stockholm till Berlin vinner. Yngve och Zeke tillhör samma lag och har båda gått 14300 steg den 25 maj. De rapporterar in resultaten samtidigt. När båda gjort sina rapporteringar har lagets antal endast ökat med 14300 och inte 28600 som väntat. Vad är det som har hänt och hur ska serverprogammet åtgärdas för att fortsatta rapporteringar ska bli rätt?

4. (3p) Nedan följer fem korta beskrivningar av designmönster. Välj ut vilken beskrivning som stämmer för var och en av

Decorator **Mediator** **Proxy**

A) Skjut upp delar av en algoritm till subklasser.
B) Ett objekt kontrollerar åtkomsten till ett annat objekt. De båda implementerar samma gränssnitt.
C) Ett objekt har hand om kommunikationen mellan n st objekt.
D) Lägg till funktionalitet till ett objekt dynamiskt. Javas klasser för strömmar följer mönstret.
E) En enda klass ger ett gränssnitt till ett komplext system.

5. (2p) **a)** Vad är namnet på det designmönster där ett objekt underrättar n st andra objekt när det första objektets tillstånd ändras?

b) Förklara kort hur mönstret tillämpas på Javas grafiska komponenter och lyssnare.

6. (2p) Para ihop arbetssätten *Vattenfall* och *eXtreme Programming* med följande karakteristika:

A) Testfasen ligger sent i projektet
B) Programmeringen sker i par
C) Systemspecifikation görs tidigt
D) Kunden/beställaren är med hela tiden i projektet

7. (1p) Vilket av följande alternativ beskriver bäst termen **refactoring** ?

A) Uppdelning av programkoden i många små delar (klasser och metoder).
B) Förbättring av programkoden utan att programmets yttre funktion ändras.
C) Upprepad testning av ett program allteftersom det byggs ut.
D) Förbättring av det gränssnitt ett program visar mot användaren med så lite förändringar av programkoden som möjligt.

8. (1p) Antag att `C1` och `C2` är klasser och att `I1` och `I2` är interface (allt i Java förstås). Vilket av följande sätt är **inte** en tillåten inledning på en ny deklaration av klass eller interface ?

A) `class C extends C1, C2`
B) `class C implements I1, I2`
C) `class C extends C1 implements I1`
D) `interface I extends I1, I2`

9. (2p) Vad blir utskriften från programmet?
Svarsalternativ finns efter programkoden. (1p)
Förklara hur du kom fram till ditt svar. (1p)

```
class Fraga9 {
    public static void main (String[] u) {
        Person pelle = new Person();
        Person samanta = new Motionär();
        Motionär anna = new Motionär();

        pelle.aktivitet();
        samanta.aktivitet();
        anna.aktivitet();
        System.out.println();
    }
}

class Person {
    void  aktivitet () {
        System.out.print("Vila ");
    }
}

class Motionär extends Person {
    void  aktivitet () {
        System.out.print("Träna ");
    }
}
```

A) Vila Vila Träna
B) Vila Träna Träna
C) Vila Vila Vila
D) Träna Träna Träna

10. (2p) Lös koppling mellan programdelar (klasser) eftersträvas ofta i objektorienterad programmering. Om klassen `A` beror av klassen `B`, hur kan man lösa upp detta beroende och åstadkomma en lösare koppling mellan klasserna `A` och `B`?

Svaret kan ges i Javakod eller i UML-klassdiagram.

```
class A {
  B minB;
  ...
}

class B {
   <B:s många variabler och metoder>
}
```

11.(1p) Vad är **JUnit** ?

A) Klass i Java API:n med fysikaliska konstanter i SI-enheter.
B) Paket med Java-komponenter speciellt anpassade för Unix-miljön.
C) *Java Junior training* - speciell utvecklingsmiljö för att lära barn programmera i Java.
D) Java-ramverk för automatiserad testning som används bl.a. inom XP.

del II

12. (16p) Här ska en grafisk komponent som är liknar en himmel med tindrande stjärnor skissas. Komponenten ärver från Canvas eller JPanel. För att tindrandet ska kunna ske oberoende av komponentens omgivning görs det i en egen tråd. Tindrandet åstadkoms genom att stjärnorna ritas om med jämna mellanrum och genom att ca 20% av stjärnorna ges ändrad diameter före omritningen. Det behöver inte vara samma stjärnor som ändras i varje omritning.
Delfrågorna a) – e) kan besvaras oberoende av varandra!

a) (1p) Hur åstadkoms komponentens egen tråd ? Svara kort, *två ord* räcker!

Antag att stjärnhimmelklassen heter `Sky` och att stjärnorna representeras av många objekt av klassen `Star` som är en inre klass i `Sky`.

```
class Star {
    int x, y;
    double diameter;
    Color col;

    Star (...) {...} // konstruktor

    void draw (...) {...} // ritar stjärnan i komponenten i
                          // position (x,y) och med aktuell diameter

    void changeDiameter () {...} // ändra diametern lagom mycket
                                 // för tindrandet
    // fler metoder
}
```

Stjärnobjekten är skapade och finns i listan `ArrayList<Star> allStars`.

b) (6p) Skriv den del av klassen `Sky` som med jämna mellanrum ritar om komponenten och ser till att den ”tindrar”. Du kan ha användning för metoden `Math.random()` som vid anrop ger ett slumptal på intervallet $[0, 1[$ samt av metoden `Thread.sleep(ms)` som låter tråden vänta `ms` millisekunder. Du behöver inte bry dig om vilka färger som används utan anta att bakgrundsfärgen är satt till något mörkt och att metoden `draw()` i `Star` är allt som behövs för att rita en stjärna. All syntax behöver inte vara rätt för att få full poäng.

c) (3p) (p) Du ser flera möjliga utvidgningar av den tindrande stjärnhimlen: En variant är att låta stjärnorna ändra färg i samband med att de tindrar. En annan är att låta stjärnor försvinna om de blivit små och att låta nya ”födas”. Du kommer fram till att du kan definiera och anropa en eller flera *tomma* metoder i `Sky` som senare kan fyllas med innehåll för att realisera olika utvidgningar.

Vilket designmönster är detta? (1p)
När/var ska de icke-tomma metoderna med utvidgningar definieras? (1p)
Förklara kort hur mönstret fungerar relaterat till uppgiften eller fristående från uppgiften. (1p)

d) (3p) Hur åstadkommer man att tindrandet kan stängas av och sättas på genom att man klickar med musen i komponenten ? Ledtråd: Musklick i `Canvas` eller `JPanel` genererar `ActionEvent`. Beskriv i ord eller pseudokod eller Javakod eller kombinera som du vill.

e) (3p) Rita komponentens UML-klassdiagram enligt a), b) och d). För full poäng skall du ha med i diagrammet att komponenten är *beroende* av klassen `Graphics` fast denna inte finns med i uppgiftslydelsen och inte heller behöver användas i några av lösningarna till a)–d).

13. (7p) Träningsföreningen Hurtbullen (konkurrent till F&S) samlar information om sina motionärers aktiviteter i en XML-databas. Följande DTD beskriver aktiviteterna för en enda motionär under en period av några veckor.

```
<!DOCTYPE Träning [
    <!ELEMENT Träning (Vecka*)>
    <!ATTLIST Träning id CDATA #REQUIRED>
    <!ELEMENT Vecka (Dag+)>
    <!ATTLIST Vecka nr CDATA #REQUIRED>
    <!ELEMENT Dag (Pass+)>
    <!ATTLIST Dag namn CDATA #REQUIRED>
    <!ELEMENT Pass (#PCDATA)>
    <!ATTLIST Pass webbbokad (TRUE|FALSE) "TRUE")>
]>
```

a) (6p) Skriv en XML-fil där medlemmen med id=1231 har tränat följande pass: I vecka 20 blev det Medeljympa på onsdagen och fredagen, i vecka 21 blev det Skivstång *och* Medelspinning på måndagen, samt Medeljympa på torsdagen och lördagen. Alla pass bokades på webben utom spinningen på måndagen som var en ”spontanträning”. Veckodagarnas namn kan skrivas som må, ti, on, to, fr, lö, sö. Startrad av typen `<?xml version="1.0"?>` behöver inte vara med.

b) (1p) Vad betyder * respektive + som står efter Vecka, Dag och Pass i DTD:n ?