Skolan för datavetenskap och kommunikation

Objektorienterad Programkonstruktion, DD1346

Tentamen 2017–06–10, kl. 9.00–12.00 FACIT

Tillåtna hjälpmedel: Papper, penna och radergummi.

Notera: Frågorna besvaras på separat papper. Svaren till del I kan skrivas på samma sida om de får plats, men behandla högst en uppgift per sida för del II. Kom ihåg att skriva namn och personnummer på alla inlämnade blad. Skriv tydligt!

Betygsgränser: Betyg FX: $\geq$ 17p i del I
Betyg E: $\geq$ 20p i del I
Betyg D: $\geq$ 20p i del I **och** betyg D på del II
Betyg C: $\geq$ 20p i del I **och** betyg C på del II
Betyg B: $\geq$ 20p i del I **och** betyg B på del II
Betyg A: $\geq$ 20p i del I **och** betyg A på del II

Eventuella bonuspoäng adderas till resultatet i del I.
För betyg på del II, se instruktionerna till del II.

Ansvarig: Christian Smith (ccs@kth.se)

Lycka till!

Del I - flervalsfrågor

1. I denna uppgift följer beskrivningar av 6 olika designmönster. För varje beskrivning, ange namnet på mönstret som bäst matchar beskrivningen (välj ur listan nedan). Varje korrekt angett mönster ger 1 p. (6 p)

   Factory **Lock** **Flyweight** **Proxy** **Threadpool** **Iterator** **Composite**
   Facade **MVC** **Observer** **Builder** **Singleton** **Adapter** **Prototype**

   a) Man skapar en ny klass som låter en befintlig klass använda en annan, trots att de inte är explicit kompatibla. Adapter

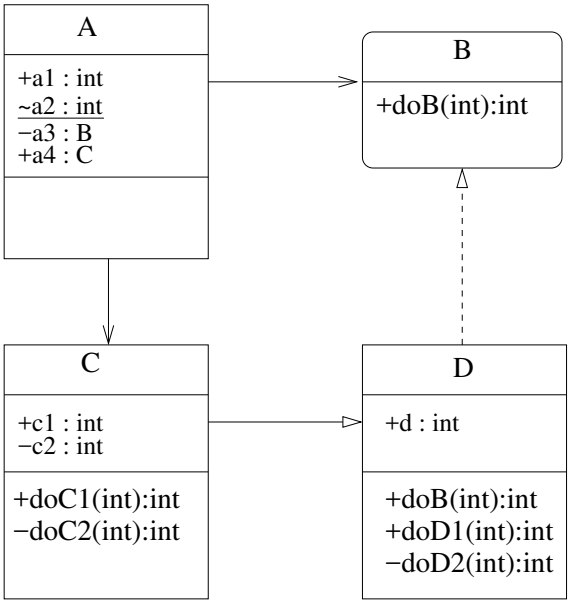
   b) Man har en klass **A** med samma API som en klass **B**, så att instanser av klass **A** används i stället för instanser av **B**, som av någon anledning inte är direkt tillgängliga. Proxy

   c) Man gör så att en resurs, t.ex en metod, bara kan användas av en tråd i taget. Lock

   d) Man skapar nya objekt med hjälp av en metod som väljer vilken typ av objekt som ska returneras utifrån argument eller yttre omständigheter, i stället för att anropa konstruktorn direkt. Factory

   e) Man skriver en ny klass **A** med ett enkelt API, som internt samlar en komplex struktur med många klasser och anrop, som till största delen döljs för användare av **A**. Facade

   f) Man har flera likartade objekt, och lagrar gemensama data på en extern plats för att spara minne. Flyweight



2. Nedan följer ett antal olika versioner av en `main()`-metod man skulle kunna skriva i klassen **A** i UML-diagrammet ovan. För varje version, ange om den kompilerar, eller om det skulle ge upphov till ett kompileringsfel. Alla klasserna finns i samma paket, och ligger i samma katalog. Varje korrekt utvärderad `main()`-metod ger 0.5 p. (4 p)

   a) kompileringsfel

   ```
   public static void main(String[] args){
       a3 = new D();
       a2 = a3.doB(1);
   }
   ```

   b) kompilerar

   ```
   public static void main(String[] args){
       D myD = new D();
       a2 = myD.doD1(1);
   }
   ```

c) kompileringsfel

```
public static void main(String[] args){
    B myD = new D();
    a2 = myD.doD1(1);
}
```

d) kompileringsfel

```
public static void main(String[] args){
    B myB = new B();
    myD.doB(1);
}
```

e) kompileringsfel

```
public static void main(String[] args){
    C myC = new C();
    D myD = new D();
    a1 = myC.doC1(myD.doD1(1));
}
```

f) kompileringsfel

```
public static void main(String[] args){
    a2 = 1;
    a3 = new C();
    a2 = a3.doC2(a2);
}
```

g) kompileringsfel

```
public static void main(String[] args){
    B myB = new C();
    D myD = (D)myB;
    C myC = (C)myD;
    myC.doC2(1);
}
```

h) kompilerar

```
public static void main(String[] args){
    A myA = new A();
    myA.a4 = new C();
    myA.a1 = myA.a4.doB(1);
    myA.a3 = myA.a4;
}
```

3. Här följer 5 st kodlistningar i Java[1]. För varje kodlistning, ange om den kommer att fungera, ge kompileringsfel eller ge fel när man kör den. Kod anses fungera **omm** den ger en deterministisk utskrift vid körning. Anta att varje klass **X** finns i en fil som heter '`X.java`', och kompileras med kommandot '`javac X.java`', och körs med '`java X`'. (5 p)

a) Fungerar

```
public class A{

    public static void main(String[] args){
        System.out.println(new A());
    }

    public String toString(){
        return "Math.random()";
    }

}
```

b) Fungerar inte (icke-deterministisk)

```
public class B{

    private static double b = 0.0;

    public static void main(String[] args){
        B b = new B();
        System.out.println(b);
    }

    public String toString(){
        return Double.toString(Math.random());
    }


}
```

[1]För tydlighets skull anses här Java 8, som finns i skolans datasalar

c)

Fungerar

```
public class C{

    static C myC = new C();

    public static void main(String[] args){
        // Do nothing!
    }

    public C(){
        System.out.println("C");
    }


}
```

d)

Fungerar

```
public class D implements Runnable{

    private int d = 0;
    static Object lock = new Object();

    public static void main(String[] args)throws InterruptedException{
        Thread th1 = new Thread(new D());
        Thread th2 = new Thread(new D());

        th1.start();
        th2.start();

        th1.join();
        th2.join();

        System.out.println(new D().d);

    }

    public void run(){
        while(d < 50000){
            incD();
        }
    }

    private void incD(){
        synchronized(lock){
            d++;
        }
    }

}
```

e)

Kompileringsfel

```
public class E extends Thread{

    public int e = 0;
    public Object lock;
    public E E1;
    public E E2;

    public static void main(String[] args) throws InterruptedException {
        E1 = new E(new Object());
        E2 = new E(new Object());

        E1.start();
        E1.join();
        E2.start();
        E2.join();

        System.out.println(e);
    }

    public void run(){
        doStuff();
    }

    public E(Object o){
        lock = o;
    }

    synchronized private void doStuff(){
        while(++e < 50000){
            synchronized(lock){
                e += e++;
            }
        }
    }
}
```

4. För varje påstående om nyckelord i Java, ange om det är sant eller falskt. 5 korrekta svar ger 4p, 4 korrekta svar ger 2p, 3 korrekta svar ger 1p, 2 eller färre korrekta svar ger 0p. (4 p)

   a) Ett fält som deklarerats som **static** kommer automatiskt att uppdateras i alla objekt av en viss klass om dess värde ändras i något objekt av samma klass. Sant

   b) En metod som deklarerats som **abstract** kan inte överskuggas i en ärvande klass. Falskt

   c) En klass som har deklarerats som **final** kan inte instansieras. Falskt

   d) En metod som deklarerats som **protected** kan inte överskuggas i en ärvande klass. Falskt

   e) Ett fält som deklarerats som **public** kan ändras av metoder från andra klasser. Sant

5. För varje påstående, ange om det är sant eller falskt. 5 korrekta svar ger 4p, 4 korrekta svar ger 2p, 3 korrekta svar ger 1p, 2 eller färre korrekta svar ger 0p. (4 p)

   a) Om man försöker starta fler trådar än det finns processorkärnor uppstår **livelock**. Falskt

   b) Ett sätt att undvika **deadlock** är att bara anropa metoder som deklarerats som **synchronized**. Falskt

   c) I Java kan olika trådar dela minne, och använda samma objekt. Sant

   d) **Thread interference** kan uppstå om två trådar arbetar med samma resurs samtidigt. Sant

   e) **Starvation** är när man skapar för många trådar så att arbetsminnet tar slut. Falskt

6. För varje påstående om UML, ange om det är sant eller falskt. 4 korrekta svar ger 2p, 3 korrekta svar ger 1p, 2 eller färre korrekta svar ger 0p. (2 p)

   a) Klassdiagram får inte innehålla gränssnitt. Falskt

   b) Alla relationer i klassdiagram måste alltid ha en riktning. Falskt

   c) Eftersom samma piltyper används för att visa relation till en superklass som till ett implementerat gränssnitt, måste gränssnitt alltid märkas med `<interface>` för att undvika missförstånd. Falskt

   d) Privata fält märks med en tilde (~) i klassdiagram. Falskt

Del II - fördjupningsfrågor

Följande 5 frågor kan betygsättas med antingen **A** eller **C**. Betyget **A** ges då allt som efterfrågas i uppgiften besvaras korrekt, med en tillräcklig motivering. Mindre detaljfel som inte påverkar helheten kan accepteras. Betyget **C** ges då någon del av uppgiften inte besvarats korrekt eller getts tillräcklig motivering, eller om texten utöver efterfrågat svar innehåller betydande mängd information som inte efterfrågats (dvs. man får inte "dubbelgardera"). Om betydande del av efterfrågat svar saknas eller är otillräckligt motiverat ges inget betyg.

Följande resultat krävs för att uppnå respektive betyg på denna del:

Betyg D: Uppnått lägst betyg **C** på minst 2 uppgifter.
Betyg C: Uppnått lägst betyg **C** på minst 4 uppgifter.
Betyg B: Uppnått betyg **A** på minst 2 uppgifter och **C** på alla resterande.
Betyg A: Uppnått betyg **A** på minst 4 uppgifter och **C** på eventuell resterande.

Besvara varje uppgift på ett separat papper.

7. Vad är en **Threadpool**? Beskriv hur och varför den används, och hur den kan implementeras.

8. Vad är en **Remote Proxy**? Beskriv hur och varför den används, och hur den kan implementeras.

9. Vad är en **klassmetod**, och hur skiljer den sig från en **instansmetod**? När bör man använda klassmetoder? Varför?

10. Du har märkt att väldigt många av dina programmeringsprojekt kräver att man sparar loggar av någon sort. Det kan röra sig om resultat från simulerade experiment (stora serier av numeriska data), chat-konservationer (texter med tillhörande metadata, som tid, person, samtalspartners, etc), felloggar (nogranna tekniska beskrivningar av olika programtillstånd så att man kan spåra och rätta till fel), mm. I stället för att uppfinna hjulet på nytt varje gång bestämmer du dig för att skriva ett generiskt loggbibliotek. Beskriv hur du skulle gå till väga för att skriva ett lättanvänt loggbibliotek som kan hantera alla möjliga typer av loggar som kan tänkas kunna uppstå i framtida projekt. Namnge alla ingående designmönster korrekt och motivera dina designval väl.

11. En vän sommarjobbar på ett hemligt programmeringsprojekt, och söker din hjälp. Hen har skrivit ett stort och avancerat programpaket, som när det ska köras på målplattformen visar sig köra alldeles för långsamt för att vara användbart. Det är dock fortfarande några veckor kvar att fixa till problemen innan sommaren är slut. Du föreslår att hen ska använda profileringen i en IDE, t.ex Netbeans, för att hitta flaskhalsar.

    Tyvärr finns det inga profilerare installerade på systemet som används, och vännen får varken installera ny programvara på jobbet, eller ta med sig koden utanför arbetsplatsen. Beskriv hur hen ska gå till väga för att lösa sitt problem inom den befintliga utvecklingsmiljön (kodeditor och kompilator). Beskriv möjliga fallgropar, och ge förslag till hur de kan undvikas.