DUGGA: Objektorienterade applikationer

Läs detta!

- *Uppgifterna är inte avsiktligt ordnade efter svårighetsgrad.*
- Skriv ditt namn, personnummer och e-postadress på försättsbladet.
- **Skriv rent dina svar. Oläsliga svar r ä t t a s e j!**
- Programmen skall skrivas i Java 5, eller senare version, vara indenterade, renskrivna och i övrigt vara utformade enligt de principer som lärts ut i kursen.
- Onödigt komplicerade lösningar ger poängavdrag.
- Givna deklarationer, parameterlistor, etc. får ej ändras, såvida inte annat sägs i uppgiften.
- Läs igenom tesen och förbered ev. frågor.

I en uppgift som består av flera delar får du använda dig av funktioner klasser etc. från tidigare deluppgifter, även om du inte löst dessa.

Lösningarna fylls i direkt i tesen.

Lycka till!

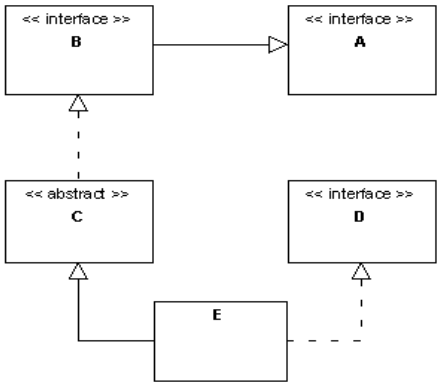
Uppgift 1 Modellering

a) Rita ett UML-diagram som motsvarar följande Java-kod. Det räcker om du ritar klassikonerna med klassnamnen ifyllda samt lämpliga relationspilar mellan dem.

```
public class A {
    private int size;
    public A(int size) { this.size = size; }
    ...
}
public class B {
    private List<A> as;
    public B() { as = new ArrayList<A>(); }
    public void add(A a) { as.add(a); }
    ...
}
public class C {
    private B b;
    private String name;
    public C(B b,String name) { this.b = b; this.name = name; }
}
public class Main {
    public static void main(String[] arg) {
        B theB = new B();
        for ( int i=0,int s=1; i < Integer.parseInt(arg[0]); i++) {
            theB.add(new A(s));
            s = s * 2;
        }
        List<C> cs = new ArrayList<C>();
        for ( int i = 0; i < Integer.parseInt(arg[1]); i++)
            cs.add(new C(theB,”client” + i));
        ...
    }
}
```

(3 p)

b) Skriv klassdeklarationer i Java som motsvarar följande UML-diagram. Det räcker med klasshuvudena.



(2 p)

Uppgift 2 Grafiska gränssnitt

Komplettera följande ofullständiga GUI-kod. När man skapar ett objekt av klassen `Gui` skall ett fönster visas. I fönstret skall finnas en knapp med texten `"Update"` och ett textfält av typen `JTextField`. När man trycker på knappen skall metoden

```
private String computeSomething()
```

anropas och returvärdet skrivas i textfältet. Textfältet skall ej vara editerbart för användaren.

```
public class Gui
{


    public Gui() {  makeFrame(); }

    private void makeFrame() {









    }

    private String computeSomething() { ... } // given
}
```

(5 p)

Uppgift 3 Designmönster

När klassn `Gui` nedan instansieras får man upp ett litet fönster med en knapp med texten `Roll the die` (kasta tärningen) och ett textfält där tärningens värde visas.

När man trycker på knappen skall tärningens nya värde visas i textfältet. Klasserna är delvis färdiga, men det återstår en sak att göra, nämligen att tillämpa designmönstret Observer så att `Gui` blir observatör till `Die`. Komplettera koden!

```
public class Die
{
    private int value;
    private static Random rand = new Random();

    public Die() { value = nextInt(); }

    public void roll() {
        value = nextInt();



    }
    public int getValue() { return value; }
    private int nextInt() { return rand.nextInt(6) + 1; }
}

public class Gui
{
    private Die die;
    private JTextField textField;

    public Gui() {
        makeFrame();
        die = new Die();


    }
    private void makeFrame() {  // Färdig metod.
        // Metoden adderar en knapp och ett textfält till fönstret.
        // En lyssnare anropar die.roll() vid tryck på knappen.
    }
```

```
}
```

(5 p)

Uppgift 4 Aktiva objekt

Komplettera följande klass så att ett klockslag visas i en komponent av typen `JLabel` varje sekund. Använd en Swing-timer. Ex. När exekveringen startar skall tiden `00:00:00` visas i komponenten, efter 17 sekunder `00:00:17`, och efter 4847 sekunder `01:20:47`, etc. Använd den färdiga metoden

```
private String formatClockTime(int seconds)
```

för att formatera sekundtid till klocktid i strängform. Metoden returnerar en sträng enligt exemplen ovan.

```
public class Clock
{




    public Clock() {  makeFrame(); }

    private void makeFrame() {









    }









    private String formatClockTime(int seconds) { ... } // Given
}
```

(5 p)

Uppgift 5 Strömmar och filer

Du har en lista av mätvärden av typen `Double` och vill skriva dessa till en binär fil vars namn ges som argument till programmet. Först skall antalet tal i listan skrivas till filen och därefter talen. Vilka strömklasser bör användas? Skriv färdigt programmet!

```
public class Main {
    public static void main(String[] arg) {
        List<Double> values = new ArrayList<Double>();
        ...
        String fileName = arg[0];









    }
}
```

(5 p)

Uppgift 6 Kommunikation

Du håller på att bygga upp en Java-klass med olika hjälpmedel för att underlätta kommunikation. Klassen skall bl.a. innehålla metoden `getRemote` som hämtar en kort text från en annan dator. Den skall ta en URL i textform, läsa filen, samt returnera innehållet i form av en sträng. Du kan anta att filen som skall läsas endast innehåller en textrad. Observera att metoden ej själv skall fånga några undantag.
Ex.

```
String content;
try {
    content = getRemote("http://www.banken.se/mitt_konto.txt");
}
catch (...) {...}
```

Skriv färdigt metoden!

```
public static String getRemote(String urlString)
throws MalformedURLException,IOException
{




}
```

(5 p)