

CHALMERS

Institutionen för data- och informationsteknik

TENTAMEN

KURSNAMN	Objektorienterad programmering, 7.5p
PROGRAM LÄSÅR	D2 2008/2009, lp 1
KURSBETECKNING	DAT041
EXAMINATOR	Uno Holmer
TID FÖR TENTAMEN	Fredagen den 28/8 2009, 8.30-12.30
HJÄLPMEDEL	Java API (21 numrerade sidor)
ANSV LÄRARE DATUM FÖR ANSLAG	Jan Skansholm tel. 772 1012 Senast den 21/9 2009 datum för visning meddelas senare
ÖVRIG INFORM.	Betygsgränser: 3 - 24p, 4 - 36p, 5 – 48p. (max 60p)

Vissa uppgifter kan besvaras direkt i tesen

Idnr:

Kom ihåg att lämna in den!



TENTAMEN: Objektorienterad programmering

Läs detta!

- *Uppgifterna är inte ordnade efter svårighetsgrad.*
- Börja varje uppgift på ett nytt blad.
- Skriv ditt idnummer på varje blad (så att vi inte slarvar bort dem).
- **Skriv rent dina svar. Oläsliga svar räknas ej!**
- Programkod som finns i tentamenstesen behöver ej upprepas.
- Programkod skall skrivas i Java 5 (eller senare) och vara indenterad och kommenterad.
- Onödigt komplicerade lösningar ger poängavdrag.
- Givna deklARATIONER, parameterlistor etc. får ej ändras.
- Läs igenom tentamenstesen 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.
--

Lycka till!

Uppgift 1

Välj ett alternativ för varje fråga! Garderingar ger noll poäng. Inga motiveringar krävs. Varje korrekt svar ger två poäng. *Besvara direkt i tesen!*

1. Givet klassen Num

```
public class Num {  
    private int x;  
    public Num() { x = 0; }  
    public void set( int x ) { this.x = x; }  
    public int get() { return x; }  
}
```

vad skriver kodavsnittet nedan ut?

```
ArrayList<Num> list = new ArrayList<Num>();  
Num n = new Num();  
  
for ( int j = 0; j < 3; j++ )  
    list.add( n );  
  
int k = 1;  
for ( Num x : list )  
    x.set( k++ );  
  
for ( Num x : list )  
    System.out.println( x.get() );
```

- a. 0 0 0
- b. 3 3 3
- c. 1 2 3

2. Vinner Kalle millionen?

```
String name;  
... // Kalle skriver sitt namn (och stavar rätt)  
    // som läses in till variabeln name  
  
if ( name == "Kalle" )  
    System.out.println("Grattis, du har vunnit en million");
```

- a. Ja, helt klart!
- b. Nej, han kan känna sig blåst.
- c. Kanske, med lite tur...

3. Endast en av deklarationerna är typkorrekt, vilken?

- a. `LinkedList<LinkedList> l = new List<LinkedList>();`
- b. `List<List> l = new ArrayList<List>();`
- c. `List<LinkedList> l = new LinkedList<List>();`
- d. `LinkedList<LinkedList> l = new LinkedList<List>();`

4. I vilken ordning bör nedanstående aktiviteter genomföras vid utbyggnad av programkod med mer funktionalitet?
- a. refaktorera, regressionstesta, bygg ut, regressionstesta
 - b. regressionstesta, refaktorera, bygg ut
 - c. regressionstesta, bygg ut, refaktorera
 - d. bygg ut, regressionstesta, refaktorera, regressionstesta

5. Vilket påstående stämmer bäst?

```
public class Person {  
    private String name;  
    private String address;  
    private String phone;  
    private String email;  
  
    // date of last visit  
    private int year;  
    private String month;  
    private int day;  
  
    ... // metoder utelämnade  
}
```

- a. Person har låg kohesion
- b. Person har hög kohesion
- c. begreppet kohesion är enbart tillämplbart på metoder, inte på klasser.

(10 p)

Uppgift 2

Följande program kan användas för att konvertera från dollar (\$) till svenska kronor (SEK).



a) Utvidga programmet med ett menyalternativ för att avsluta exekveringen. Exekveringen av ett Java-program kan avslutas med anropet `System.exit(0)` ;

(5 p)

b) Lägg till konvertering från svenska kronor till dollar.

(5 p)

Fyll i dina lösningar nedan.

```
public class CurrencyConverter {
    public final static double dollarToSek = 7.81;
    private JFrame frame;
    private JLabel sekDigits;
    private JLabel dollarDigits;

    public CurrencyConverter() { makeFrame(); }

    private void dollarsToSek() {
        String digits =
            JOptionPane.showInputDialog(null,
                                       "Type in the price in $",
                                       "Currency converter",
                                       JOptionPane.PLAIN_MESSAGE);

        if ( digits == null )
            return;
        Double dollarValue = Double.parseDouble(digits);
        dollarDigits.setText(dollarValue.toString());
        Double sekValue = dollarToSek*dollarValue;
        sekDigits.setText(sekValue.toString());
    }
}
```

```
private void makeFrame() {
    frame = new JFrame("Currency converter");
    makeMenuBar(frame);

    JPanel contentPane = (JPanel)frame.getContentPane();
    contentPane.setLayout(new GridLayout(2,2));
    sekDigits = new JLabel();
    dollarDigits = new JLabel();

    contentPane.add(new JLabel("$"));
    contentPane.add(dollarDigits);
    contentPane.add(new JLabel("SEK"));
    contentPane.add(sekDigits);

    frame.pack();
    frame.setVisible(true);
}

private void makeMenuBar(JFrame frame) {
    JMenuBar menubar = new JMenuBar();
    frame.setJMenuBar(menubar);

    JMenu menu = new JMenu("Actions");
    menubar.add(menu);

    JMenuItem dollarsToSekItem = new JMenuItem("Dollars to SEK");
    dollarsToSekItem.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) { dollarsToSek(); }
    });
    menu.add(dollarsToSekItem);
}
```

Uppgift 3

a) Vad skrivs ut av följande metodanrop? Motivera!

```
Base obj = new Sub();
obj.f1();           // 1.
obj.f2();           // 2.
obj.f3();           // 3.

Sub obj2 = new Sub();
obj2.f1();          // 4.
obj2.f2();          // 5.
obj2.f3();          // 6.
obj2.f4();          // 7.

public class Base {
    public void f1() { System.out.print("Base.f1"); }
    public void f2() { System.out.print("Base.f2"); }
    public void f3() { System.out.print("Base.f3"); }
}

public class Sub extends Base {
    public void f1() { System.out.print("Sub.f1"); }
    public void f2() { super.f2(); System.out.print("++Sub.f2"); }
    public void f4() { System.out.print("Sub.f4"); }
}
```

(7 p)

b) Vilka av raderna 1-4 är tillåtna och vilka ger kompileringsfel? En av klasserna går ej att kompilera, vilken? Motivera svaren!

```
Int  obj1 = new Int();           // 1
Base obj2 = new Base();          // 2
Sub1 obj3 = new Sub1();          // 3
Sub2 obj4 = new Sub2();          // 4

public interface Int {
    public void h();
}

public abstract class Base implements Int {
    public abstract void f();
    public void g() { ... }
}

public class Sub1 extends Base {
    public void f() { ... }
}

public class Sub2 implements Int {
    public void h() { ... }
}
```

(4 p)

Uppgift 4

En programmerare fick i uppdrag att implementera en enkel adressbok i Java. Ett första försök gav följande preliminära klass:

```
public class AddressBook {  
    ... datarepresentationen utelämnad här (lista eller HashSet)  
    public int size() { ... }  
    public void add(AddressBookEntry e) { ... }  
    public boolean find(AddressBookEntry e) { ... }  
    public void remove(AddressBookEntry e) { ... }  
}
```

Klassen AddressBookEntry definieras:

```
public class AddressBookEntry {  
    private String name;  
    private String address;  
    private String email;  
    private String phone;  
  
    ... metoderna utelämnade  
}
```

I ett försök att testa adressboken skrevs följande kodavsnitt för att se om det går att byta adress för en viss person (... = detaljerna utelämnade) :

```
AddressBook b = new AddressBook();  
b.add(new AddressBookEntry("Nisse Karlsson", "... ", "...", "..."));  
b.add(new AddressBookEntry("Lisa Jonsson", "Kortstigen 2",  
    "...", "..."));  
b.add(new AddressBookEntry("Allan Molin", "a", "...", "..."));  
  
AddressBookEntry e =  
    new AddressBookEntry("Lisa Jonsson", "Kortstigen 2",  
        "...", "..."); // samma som ovan  
if ( b.find(e) ) {  
    b.remove(e);  
    b.add(new AddressBookEntry("Lisa Jonsson", "Långgatan 3",  
        "...", "..."));  
    System.out.println("Book updated");  
} else  
    System.out.println("Not found");
```

Tyvärr fungerar det inte, man får utskriften Not found, oavsett om adressboken internt implementeras med en lista eller med ett HashSet.

- a) Varför fungerar inte ovanstående lösning? (3 p)
- b) Inför nödvändiga metoder i AddressBookEntry så att adressboken fungerar oavsett om posterna lagras i en lista eller i ett HashSet. (4 p)
- c) Addera metoden clone till AddressBookEntry och AddressBook. Tekniken med kopieringskonstruktor i den senare ger full poäng, denna konstruktor skall då givetvis också införas i klassen. (4 p)

Uppgift 5

Ibland är det viktigt att klasser inte instansieras mer än en gång. En instans kan t.ex. administrera en gemensam resurs. Om flera instanser tillåts hantera resursen kan det vara svårt att garantera att detta görs på korrekt sätt. I nedanstående exempel tänker vi oss att klassen `Resource` är en sådan resurshanterare, och klasserna `Client1` och `Client2` är två användare av `Resource`. Problemet med designen är att koden i klientklasserna inte vet om de använder samma resursobjekt, vilket det borde vara, eller om de är två olika objekt. Tillämpa designmönstret Singleton på koden genom att stryka, ändra och lägga till kod på lämpliga ställen nedan.

```
public class Resource {

    public Resource() {

    }

    public void someMethod() { ... }
}

public class Client1 {
    private Resource r;

    public Client1(Resource r) {
        this.r = r;
    }

    public void clientMethod() {
        //...
        r.someMethod();
        //...
    }
}

public class Client2 {
    private Resource r;

    public Client2(Resource r) {
        this.r = r;
    }

    public void anotherClientMethod() {
        //...
        r.someMethod();
        //...
    }
}

public class Main {
    static public void main(String[] arg) {
        Resource r = new Resource();
        Client1 c1 = new Client1(r);
        Client2 c2 = new Client2(r);
        // ...
    }
}
```

Uppgift 6

Följande gränssnitt definierar metoder för enkla köer

```
public interface Queue {  
    // Adds x last in the queue.  
    void add(String x);  
  
    // Determines if the queue is empty.  
    boolean isEmpty();  
  
    // Removes and returns the first element in the queue.  
    String get() throws NoSuchElementException;  
  
    // Empties the queue.  
    void clear();  
}
```

Antag att man vill kunna addera funktionalitet så att köer kan sparas i filer, d.v.s. bli *persistenta*. Inför klassen `PersistentQueue`. Klassen skall implementera `Queue` och dessutom ha metoderna

```
// Saves the contents of this queue into a file with the specified name.  
// The queue is not altered.  
public void save(String fileName) throws IOException  
  
// Loads the contents from a file with the specified name  
// into this queue and returns a reference to it.  
public Queue load(String fileName) throws IOException
```

Med designmönstret Decorator kan problemet lösas, utan att ärva från någon konkret köklass.

- Implementera klassen `PersistentQueue` i Java enligt decoratormönstret. (7 p)
- Ge ett kodexempel som skapar en persistent kö, adderar några element, samt sparar kön i en fil. Du kan anta att klassen `SomeQueue` implementerar `Queue`, samt att den är serialiserbar. (3 p)