Instuderingsuppgifter läsvecka 3

1.

Vad är funktionell dekomposition? Vilka fördelar finns det med funktionell dekomposition?

2.

Vad innebär att en metod skall följa *The Separation of Concerns Principle*? Vilka fördelar fås?

3.

Vad är en sidoeffekt? Vad är en oönskad sidoeffekt?

4.

Vad innebär T*he Command-Query Separation Principle*?

5.

Förklara vad *kontraktbaserad design* (*Design by Contract*) innebär. Vad är ett kontrakt? Vem är ansvarig för de olika delarna i kontraktet? Vilka fördelar finns med kontraktbaserad design?

6.

Hur skiljer sig *design by contract* från *defensiv programmering*?

7.

När skall klassinvarianterna kontrolleras?

8.

Eftersom klienten ansvarar för att förvillkoren är uppfyllda måste dessa kunna verifieras mot klassens publika gränssnitt. Måste eftervillkor och klassinvarianter också vara verifierbara mot det publika gränssnittet? Motivera ditt svar!

9.

Vad har en programutvecklare för praktisk nytta av *assertions*?

10.

Koden i metoden chooseSupplier är mindre bra.

```
public static void chooseSupplier(Object obj) {
   if (o instanceof SupplierA) {
      ((SupplierA) obj).doIt();
   } else if ( obj instanceof SupplierB) {
      ((SupplierB) obj).doIt();
   } else if ( obj instanceof SupplierC) {
      ((SupplierC) obj).doIt();
   }
}//chooseSupplier

public class SupplierA {
   public void doIt() {
      System.out.println("Done by supplier A");
   }//doIt
}//SupplierA
```

```
public class SupplierB {
   public void doIt() {
      System.out.println("Done by supplier B");
   }//doIt
}//SupplierB

public class SupplierC {
   public void doIt() {
      System.out.println("Done by supplier C");
   }//doIt
}//SupplierC
```

a) Motivera vad som är problemet.

b) Lös problemet. Om du har flera lösningsalternativ, motivera varför du valt den lösning du gjort. Du får ändra fritt i koden bara resultatet blir som det ursprungliga (d.v.s. klasserna skall finnas och anropet chooseSupplier(...) skall producera samma utskrift).

11.

Förklara varför metoden setPaymentAmount i klassen CreditCardPayment bryter mot *Liskov Substitution Principle*.

```
public abstract class Payment {
   /**
      @pre amount >= 0
   **/
   public void setPaymentAmount(int amount) { . . . }
}//Payment

public class CreditCardPayment extends Payment {
   /**
      @pre  amount > = 250
   **/
   public void setPaymentAmount(int amount) { . . . }
}//CreditCardPayment
```

12.

Betrakta följande abstraktioner:

a)

```
/**
   @returns  0 if arr is null or arr is empty, otherwise the minimum value of arr
**/
public int min (int[ ] arr)
```

b)

```
/**
   @returns  the minimum value of arr
   @throws  throws NullPointerException if arr is null and EmptyException if arr is empty
**/
public int min (int[ ] arr) throws NullPointerException, EmptyException
```

c)

```
/**
   @pre arr is not null and arr is not empty
   @returns  the minimum value of arr
**/
public int min (int[ ] arr)
```

Är någon av abstraktionerna att föredra framför de andra, eller är de likvärdiga? Motivera ditt svar!

13.

a) Förklara vad det betyder att den interna representationen i en klass exponeras!

b) Vilka förändringar måste göras i klassen Line nedan, för att förhindra exponering av representationen?

```
import java.awt.Point;
public class Line {
   private Point startPoint;
   private Point endPoint;
   public Line(Point startPoint, Point endPoint) {
      this.startPoint = startPoint;
      this.endPoint= endPoint;
   }
   public Point getStartpoint() {
      return startPoint;
   }
   public Point getEndpoint() {
      return endPoint;
   }
}//Line
```

14.

Ett objekt som är en instans av en *icke-muterbar* klass förändrar inte sitt tillstånd under sin livstid. Redogör för vilka fördelar det finns med icke-muterbara objekt.

15.

Redogör för hur tolkningen är av konstruktionen **final** när den används tillsammans med

a) en klass
b) en metod
c) en parameter
d) en instansvariabel

16.

Hur åstadkommer man att en klass blir icke-muterbar?

17.

En av de viktigaste principerna vid programdesign är *The Single Responsibility Principle.* Förklara vad denna princip innebär! Vad finns det för samband mellan *kohesion*, *koppling* och *The Single Responsibility Principle*?

18.

Det publika gränssnittet för en klass skall vara konsistent. Förklara vad det betyder.

19.

Förklara med ett exempel vad principen *Information Expert Pattern* innebär.

20.

Essensen i *Law of Demeter* beskrivs ofta med parollen *Don't talk to strangers, talk only to your immediate friends*. Förklara *Law of Demeter* och hur denna paroll kommer in i bilden!

21.

Förklara innebörden av *The Interface Segregation Principle*.