Instuderingsuppgifter läsvecka 7 - LÖSNINGAR

1.

a) Att en klass inte är trådsäker innebär att flera trådar, som använder ett objekt av klassen, samtidigt tillåts förändra tillståndet hos objektet.

I klassen Product är inte metoderna addToStock och removeFromStock trådsäkra. Följande kan inträffa:

Anta att instansvariabeln nrInStock, som anger antalet produkter i lager, har värdet 12 när två trådar t1 och t2 samtidigt exekverar metoden addToStock respektive removeFromStock. I beräkningen av satserna

```
nrInStock = nrInStock + quantity;
```

respektive

```
nrInStock = nrInStock - quantity;
```

läser både t1 och t2 av värdet 12 på instansvariabeln nrInStock. Således händer följande:

t1 adderar 20 till värdet 12 och får resultatet 32

t2 subtraherar 10 från värdet 12 och får resultatet 2

t2 lagrar värdet 2 i instansvariabel nrInStock

t1 lagrar värdet 32 i instansvariabeln nrInStock

Effekten blir att instansvariabeln nrInStock nu har värdet 32, men det faktiska antalet produkter i lager är 22.

b) Mutator-metoderna addToStock och removeFromStock måste synkroniseras:

```
public synchronized void addToStock(int quantity) {
    nrInStock = nrInStock + quantity;
}// addToStock
public synchronized void removeFromStock(int quantity) {
   if (quantity > nrInStock)
      throw new IllegalArgumentException("To big quantity");
    nrInStock = nrInStock - quantity;
}// addToStock
```

2.

a)

- wait() – blockerar den anropande tråden samt låser upp den aktuella monitorns lås så att andra trådar kan gå in i monitorn. Monitorn låses igen innan tråden exekverar vidare efter anropet.
- notify() – gör att en tråd som väntar i ett anrop av wait() i den aktuella monitorn görs körbar och lämnar anropet av wait() så fort som monitorn inte längre är låst.
- notifyAll() – som notify(), fast den gör alla väntande trådar körbara.

b) För att någon av de nämnda metoderna ska kunna anropas måste exekveringen befinna sig inne i en metod som är deklarerad synchronized i objektet som vi anropar metoden på (eller i ett motsvarande synchronied-block).

3.

a) Ett problem Kalle kan få är att meddelanden försvinner eftersom set() skulle kunna anropas två gånger i följd av samma tråd. Ett annat problem som kan uppstå är att den läsande tråden stjäl all tillgänglig CPU-tid (busy-wait) vilket kan göra att andra trådar, t.ex. den som anropar set() aldrig får köra.

b)

```
public class Buffer {
   private String message;
   public synchronized void set(String m) {
      while ( message != null) {
         try {
            wait();
         } catch(InterruptedException e) {}
      }
      message = m;
      notifyAll();
   }//set
   public synchronized String get() {
      while ( message == null) {
         try {
            wait();
         } catch(InterruptedException e) {}
      }
      String m = message;
      message = null;
      notifyAll();
      return m;
   }//get
}//Buffer
```

4.

a) möjlig b) omöjlig c) möjlig d) omöjlig

Allmänt gäller: R måste förgå G, G måste föregå I, I måste föregå V och V måste föregå K, eftersom dessa utskrifter utförs i denna ordning av tråden som exekverar main-metoden. Vidare måste O föregå Y, eftersom dessa utskrifter utförs i denna ordning i run-metoden i tråden t1.