Institutionen för
Data- och informationsteknik
JSk

TENTAMEN
2007-01-15

OBJEKTORIENTERAD PROGRAMVARUUTVECKLING

för IT1 (TDA545)

TID 14.00-18.00

Ansvarig: Jan Skansholm, tel 7721012 eller 0707-163230

Resultat: Anslås senast 2007-02-05

Betygsgränser: Sammanlagt maximalt 60 poäng.
På tentamen ges graderade betyg:.
3:a 24 poäng, 4:a 36 poäng och 5:a 48 poäng

Hjälpmedel: Skansholm, *Java direkt med Swing*, valfri upplaga, Studentlitteratur

Inga kalkylatorer är tillåtna.

Tänk på:

- att skriva tydligt och disponera papperet på ett lämpligt sätt.
- att börja varje ny uppgift på nytt blad. Skriv endast på en sida av papperet
- Skriv ditt personnummer på *alla* blad.
- De råd och anvisningar som givits under kursen skall följas vid programkonstruktionerna. Det innebär bl.a. att onödigt komplicerade, långa och/eller ostrukturerade lösningar i värsta fall ej bedöms.

Uppgift 1) Vart och ett av följande kodexempel innehåller ett fel, som leder till att det inte går igenom kompileringen. Hitta felet och beskriv vad det beror på, samt hur man skulle kunna lösa det på ett bra sätt. Varje klass och gränssnitt antas ligga i en egen fil med ett korrekt namn. För att underlätta är raderna numrerade. Deluppgifterna är oberoende av varandra och kan vardera ge maximalt två poäng.

(6 p)

a)
```
 1 public interface Frukt {
 2    public double näringsvärde(double x, double y);
 3    public void ät();
 4 }
 5
 6 public class Äpple implements Frukt {
 7    public double näringsvärde(double x, double y) {
 8       return (x*y) / (x*y + (1-x)*(1-y));
 9    }
10 }
```

b)
```
11 public class Gurka {
12    protected int n;
13    public Gurka(int n) {
14       this.n = n;
15    }
16 }
17
18 public class Saltgurka extends Gurka {
19    public Saltgurka(int n) {
20       this.n = n;
21    }
22 }
```

c)
```
23 public interface Dörr {
24    public int bredd();
25 }
26
27 public interface Fönster {
28    public double bredd();
29 }
30
31 public class FransktFönster implements Dörr, Fönster {
32    public int bredd() {
33       return 900;
34    }
35    public double bredd() {
36       return 900.0;
37    }
38 }
```

Uppgift 2) Som du vet kan ett komplext tal beskrivas med hjälp av en realdel och en imaginärdel. Din uppgift är nu att definiera en klass i Java som beskriver begreppet komplext tal. Klassen skall (minst) ha följande konstruktorer och metoder:

- en defaultkonstruktor,
- en kopieringskonstruktor,
- en konstruktor som har en realdel och en imaginärdel som parametrar,
- två metoder som används för att avläsa ett komplext tals realdel respektive imaginärdel,
- en metod som adderar ett annat komplext tal till det aktuella komplexa talet (det aktuella komplexa talet skall alltså förändras),

- en metod som multiplicerar det aktuella komplexa talet med ett annat komplext tal (det aktuella komplexa talet skall alltså förändras). Tänk på att $i \cdot i = -1$,
- en metod med namnet `equals` som undersöker om det aktuella komplexa talet är lika med ett annat komplext tal. (När man jämför variabler av typerna **`float`** eller **`double`** bör man tänka på att värdena lagras approximativt. Någon bit kan därför vara olika för variabler som man logiskt uppfattar som "lika". När man jämför två sådana variabler bör man inte jämföra dem direkt, utan istället se om absolutbeloppet av deras skillnad är mycket litet, t.ex. 10^{-15}. I så fall kan man uppfatta variablerna som lika.),
- en metod `toString` som returnerar det komplexa talet som en text med formen `"a+b*i"` eller `"a-b*i"`, beroende på om imaginärdelen är positiv eller inte.

Det skall inte vara möjligt att utifrån avläsa eller ändra ett komplext tal direkt. Allt måste göras via konstruktorer eller metoder.

(10 p)

Uppgift 3) Antag att alla resultat från en tenta finns sparade i en textfil. Varje person står på en egen rad i filen och informationen är skriven på följande form:

```
personnummer  efternamn  förnamn  u1  u2  u3  u4  u5  u6
```

`u1`, `u2`, ... står för poängen på de enskilda uppgifterna. (Antag att bara hela poäng delas ut.) På tentan kunde man få något av betygen U, 3, 4, eller 5. För att få betyget 5 krävdes minst 48 poäng, för en 4:a minst 36 poäng och för en 3:a minst 24 poäng.

Nu vill man ha informationen i filen uppdelad så att de som tentat delas upp i fyra separata filer. De med betyget 5 skall hamna i en fil, de med betyget 4 i en annan fil, de med betyget 3 i en tredje fil och de som blev underkända i en fjärde fil. Raderna i de fyra filerna skall ha samma utseende som i den ursprungliga filen, fast sist på varje rad *skall den totala poängen för eleven också skrivas ut.* Uppgiften är att skriva ett program som gör denna uppdelning.

Namnet på den ursprungliga filen skall läsas in från kommandoraden. (Om du inte vet hur man gör detta får du, mot ett poängavdrag, läsa in filnamnet från en dialogruta.) Du får alltså *inte* i ditt program förutsätta att den ursprungliga filen har ett visst, förutbestämt namn. De fyra filer som skall skapas av programmet skall ha samma namn som den ursprungliga filen fast med tilläggen `5`, `4`, `3` resp. `u` omedelbart före punkten framför suffixet. Om den ursprungliga filen t.ex. heter `resultat.txt` så skall alltså de fyra filerna heta `resultat5.txt`, `resultat4.txt`, `resultat3.txt` resp. `resultatu.txt`. Om den ursprungliga filens namn inte innehåller någon punkt skall tillläggen `5`, `4`, `3` resp. `u` finnas sist i filnamnen. Om den ursprungliga filen t.ex. heter `resultat` så skall alltså de fyra filerna heta `resultat5`, `resultat4`, `resultat3` resp. `resultatu`.

(14 p)

Uppgift 4) Uppgiften är att skriva ett program som visar ett virtuellt tangentbord på skärmen, dvs ett program som visar en massa knappar vilka fungerar som ett vanligt tangentbord. När du kör ditt program skall det se ut som i följande figur (den nedersta långa knappen är mellanslag). Ovanför knapparna skall den inmatade texten visas. När man trycker på en av knapparna läggs dess tecken till i slutet av texten. Det går inte att sudda det som redan står i texten och det skall inte gå att skriva eller förändra texten på något annat sätt än genom att trycka på knapparna.

Det finns många knappar för att uppgiften inte skall gå att lösa med klipp-och-klistra-programmering. Försök istället hitta en mer strukturerad lösning, som inte är beroende av antalet knappar. Lösningar som innehåller "och så gör vi alla de andra knapparna på samma sätt", "och så vidare", "..." eller liknande kommer inte att godkännas, utan du måste skriva ut *hela* din lösning.

(10 p)

Uppgift 5) Ett s.k. anagram får man om man utgår från ett ord eller en mening och kastar om bokstäverna så att ett nytt ord eller en ny mening framträder. Av ordet *metodanrop* kan man t.ex. bilda anagrammet *peta ond orm*. Skriv en klassmetod `ärAnagram` som undersöker om en text innehåller ett anagram av en annan text. De två texterna ges som parametrar till metoden. Resultatet skall ha typen **`boolean`**. Tips: Innan texterna jämförs kan man ta bort alla blanka tecken och se till att endast små (eller stora) bokstäver används.

(10 p)

Uppgift 6) Som bekant kan en metod bara returnera ett enda värde, vilket ibland kan kännas begränsande. Antag t.ex. att du vill skriva en klassmetod `statistik` som har två parametrar `x` och `n`. Parametern `x` är ett fält med element av typen **`double`** och `n` är ett heltal som anger hur många av elementen i fältet `x` som har utnyttjats (dvs. fyllts i). Du vill att metoden `statistik` skall ge två resultat, dels medelvärdet $\bar{x}$ av talen i `x` och dels standardavvikelsen σ av talen. Hitta på något sätt att lösa detta problem så att `statistik` kan returnera båda talen samtidigt och skriv därefter metoden `statistik`. (Tips: Använd någon mer komplicerad typ som returtyp.) Visa också några programrader där du anropar metoden `statistik` med ett fält `a` som parameter och där du antar att man har utnyttjat `k` platser i `a`. Visa hur man efter anropet av `statistik` kan skriva ut medelvärdet och standardavvikelsen.

Följande formler är givna:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2}{n-1}}$$

(10 p)