Verónica Gaspes
ITE
21 november 2023

Inlämningsuppgifter för vecka 5

Anmärkningar

- Hur man lämnar in och vad som ska lämnas in förklaras i Blackboard. Denna dokument beskriver uppgifterna. Varje uppgift finns i en .java fil. I varje .java fil finns en kommentar som förklarar uppgifterna på engelska.
- När du löser veckans uppgifter behöver du bara använda det vi har presenterat under första, andra, tredje, fjärde och femte veckorna.
  - Många operationer och några metoder för att bygga upp uttryck i olika typer,
  - if- och switch kommandon,
  - for- och while kommandon,
  - tilldelning och System.out.println,
  - fält
  - standard input
  - omdirigering av standard input och standard output
  - statiska metoder,
  - klasser som programbibliotek
  - att rita från ett Java program
  - att använda fördefinierade abstrakta datatyper och objekt
  - datatyperna Color, Picture och String
- Inför tentan måste du kunna
  - att det finns både kommandon och uttryck
  - att det finns olika typer
  - förklara vad de kommandon vi har studerat gör
  - förklara vilka typer och värden några uttryck har
  - använda uttryck, tilldelning, System.out.println, if, switch, for och while i enkla program samt förklara vad program som är uppbyggda med dessa gör när de körs.
  - förklara hur man deklarerar, skapar, initierar och går igenom fält,
  - förklara varför programmet måste gå igenom fält för att skriva ut eller jämföra fältets element,
  - använda fält i enkla program och förklara hur programmet fungerar
  - använda standard input och omdirigering av standard input och standard output
  - vad en statisk metod i Java är och hur de kan användas för att definiera funktioner och metoder i Java
  - definiera och använda statiska metoder i ett program
  - förklara vad en klass bibliotek är samt definiera statiska metoder i en klass bibliotek
  - använda statiska metoder och klassbibliotek för att organisera program i moduler

- deklarera variabler med fördefinierade datatyper
- skapa objekt av fördefinierade datatyper
- använda objekt av fördefinierade datatyper, både föränderliga och oföränderliga, genom icke-statiska metoder
- redogöra för skillnaden mellan grundtyper och referenstyper

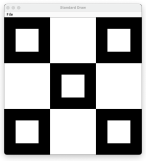
Efter inlämningsuppgifterna finns några uppgifter från gamla tentor som handlar om detta: de finns bara för att du ska kunna bekanta dig med uppgifter av tenta typ, de ska INTE lämnas in! Har du frågor om de kan du ta upp det på Drop-in passet!

Inlämningsuppgifter - del 1

1. I filen *E1.java* finns ett Java program som använder bibliotek klassen `StdDraw` (filen *StdDraw.java* följer med i zip filen). Du ska kompilera och köra programmet med sex kommandoradsargument som i

   ```
   java E1 255 200 100  200 100 255
   ```

   De första tre är RGB värdena för en färg och de sista tre är RGB värdena för en annan färg. Det ska öppnas ett StdDraw fönster med bilden

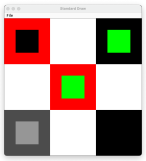
   Programmet `E1` skapar de fem rutor med en yttre och en inre ruta. Innan du löser uppgiften används bara svart och vit som färger.
   När du är klar med uppgiften ska samma kommandoradsargument visa

   Färgerna som matades in som kommandoradsargument syns i bildens mitt.
   För att använda färgerna röd (255 0 0) och grön (0 255 0) som input till programmet gör man:

   ```
   java E1 255 0 0 0 255 0
   ```

   och då blir bilden

   Det du ska göra är

   (a) Komplettera metoderna `toRed`, `toGreen`, `toBlue` i klassen `E1` enligt anvisningarna i kommentarerna direkt ovanför metod deklarationerna.

   (b) Ändra i `main` där det anvisas med kommentarer i koden som inleds med `// Your code here`:

   - i början, för att läsa in kommandoradsargumenten och skapa två färger,
   - och sedan för att använda färger enligt anvisning för alla anrop av `drawInside`. I ett av fallen ska du använda metoden `toGray` som finns redan definierad i klassen `Luminance` som följer med i zip filen.

2. I filen *E2.java* finns ett java program som använder bibliotek klassen `StdDraw` (filen *StdDraw.java* följer med i zip filen). Du ska kompilera och köra programmet som i

   ```
   java E2
   ```

Du bör se ett helt svart StdDraw fönster.
Det du ska göra är att komplettera definitionen av metoden `mix`. Metoden har signaturen

```
public static Color mix(double alpha, Color c1, Color c2)
```

Den ska räkna ut en `Color` som är en blandning av färgerna `c1` och `c2` som är argument till metoden. Hur mycket från färgen `c1` och hur mycket av `c2` som ska användas i blandningen bestäms av argumentet `alpha` som är ett flyttal mellan 0 och 1. Resultatfärgens RGB komponenter ska beräknas enligt denna formel som använder `alpha` (α) och RGB komponenterna av `c1` och `c2`:

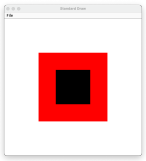
$$\begin{aligned} R &= \alpha \cdot R_{c1} + (1-\alpha) \cdot R_{c2} \\ G &= \alpha \cdot G_{c1} + (1-\alpha) \cdot G_{c2} \\ B &= \alpha \cdot B_{c1} + (1-\alpha) \cdot B_{c2} \end{aligned}$$

Programmet som finns i `main` använder metoden `mix` med färgerna svart och vit samt 10 olika värden för α: 1.0, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2 och 0.1. För varje mix ritas en smal rektangel: det blir 10 nyanser av grå, där den första är helsvart:

3. I filen *`E3.java`* finns ett Java program som använder bibliotek klassen `StdDraw` (filen *`StdDraw.java`* följer med i zip filen). Du ska kompilera och köra programmet som i

   ```
   java E3 255 0 0
   ```

   De tre heltalen står för RGB komponenterna i en färg. Programmet ritar en kvadrat i den färgen och en mindre kvadrat i färgens *komplement*. Komplementet beräknas med en metod `complement` som innan du löser uppgiften beräknar färgen svart. Du bör se ett StdDraw fönster med

   Det du ska göra är att komplettera definitionen av metoden `complement`. Metoden har signaturen

   ```
   public static Color complement(Color c)
   ```

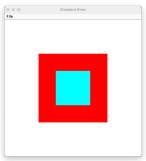
   Metoden ska beräkna en färg som är komplementet till färgen `c` som är metodens enda argument. RGB komponenterna av färgen `c` beräknas med formeln:

   $$\begin{aligned} R &= 255 - R_c \\ G &= 255 - G_c \\ B &= 255 - B_c \end{aligned}$$

   När du har löst uppgiften ska

```
java E3 255 0 0
```

resultera i



4. filen *E4.java* finns ett Java program som använder datatypen `Picture` (filen *Picture.java* följer med i zip filen). Du ska kompilera och köra programmet med

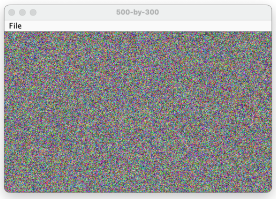
   ```
   java E4 500 300
   ```

   Du bör inte se något hända: programmet gör inget mer än att läsa in kommandoradsargumenten.

   Det du ska göra är att komplettera programmet (`main`) genom att skapa ett objekt av typen `Picture` där bredden är första kommandoradsargumentet och höjden är andra kommandoradsargumentet. Bilden ska fyllas med en slump färg i varje pixel. En slump färg är en färg där RGB komponenterna är slumptal. Programmet ska avslutas genom att visa bilden i datorns skärm.
   När du har löst uppgiften kan

   ```
   java E4 500 300
   ```

   resultera i

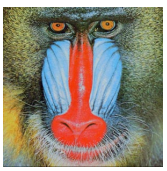


   För att avsluta programmet kan du behöva göra Ctrl-C.

5. I filen *E5.java* finns ett Java program som du ska kompilera och köra med

   ```
   java E5 baboon.jpg
   ```

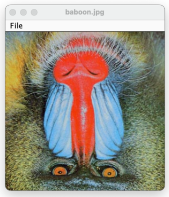
   Du bör inte se något hända: programmet gör inget mer än att läsa in kommandoradsargumentet som ska vara ett fil namn för en fil som innehåller en bild. Just filen *baboon.jpg* följer i zip filen (kan även laddas ner från kursbokens webbplats introcs.cs.princeton.edu/java/31datatype/baboon.jpg) och ser ut så här:



   Det du ska göra är att komplettera programmet (`main`) för att skapa ett objekt av typen `Picture` från bilden i filen med det namn som läses in från kommandoraden. Programmet ska då vända bilden upp-och-ner och visa bilden i datorns skärm.
   När du har löst uppgiften ska

```
java E5 baboon.jpg
```

resultera i



För att avsluta programmet kan du behöva göra Ctrl-C.

6. I filen *E6.java* finns ett Java program som du ska kompilera och köra med

   ```
   java E6 baboon.jpg
   ```

   Du bör inte se något hända: programmet gör inget mer än att läsa in kommandoradsargumentet som ska vara ett fil namn för en fil som innehåller en bild.
   Det du ska göra är

   - komplettera metoden `complement` med signatur

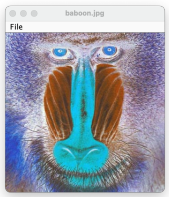
     ```
     public static void complement(Picture pic)
     ```

     som förändrar bilden `pic` (som är metodens argument) så att färgen i varje pixel blir komplementet till den färg som fanns i det pixel. I koden ska du anropa den metod för att beräkna komplementet av en färg som finns i klassen `E3`.
   - komplettera programmet (`main`) för att skapa ett objekt av typen `Picture` från bilden i filen med det namn som läses in från kommandoraden. Bilden ska förändras med metoden `complement` och visas i datorns skärm.

   När du har löst uppgiften ska

   ```
   java E6 baboon.jpg
   ```

   resultera i



   För att avsluta programmet kan du behöva göra Ctrl-C.

7. I filen *E7.java* finns ett Java program som du ska kompilera och köra med

   ```
   java E7 baboon.jpg 0.5
   ```

   Du bör inte se något hända: programmet gör inget mer än att läsa in kommandoradsargumenten. Dessa argument står för ett fil namn och en faktor. Faktorn står för hur mycket vit som ska blandas med färgen i varje pixel i den bild som finns i filen.
   Det du ska göra är

- komplettera metoden `lighter` med signatur

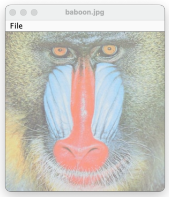
  `public static void lighter(Picture pic, double alpha)`

  som förändrar bilden `pic` (som är metodens argument) så att färgen i varje pixel blir blandad med färgen vit. Det andra argumentet, `alpha`, anger mängden vit i blandningen: när faktorn är 0 förblir färgerna oförändrade och när faktorn är 1 blir bilden vit. Blandningen ska göras med hjälp av metoden `mix` i klassen `E2`.
- komplettera programmet (`main`) för att skapa ett objekt av typen `Picture` från bilden i filen med det namn som läses in från kommandoraden. Bilden ska förändras med metoden `lighter` med andra kommandoradsargument för `alpha`.

När du har löst uppgiften ska

```
java E7 baboon.jpg 0.5
```

resultera i



För att avsluta programmet kan du behöva göra Ctrl-C.

8. I filen *E8.java* finns ett Java program som du ska kompilera och köra med

   ```
   java E8 baboon.jpg 10 0 0 0
   ```

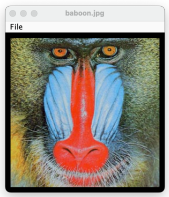
   Du bör inte se något hända: programmet gör inget mer än att läsa in kommandoradsargumenten. Dessa argument står för

   - namnet till en fil som innehåller en bild
   - bredden för en ram som ska ritas runt bilden, anges i antal pixlar
   - RGB komponenterna för ramens färg

   Det du ska göra är att komplettera programmet (`main`) för att skapa ett objekt av typen `Picture` från bilden i filen med det namn som läses in från kommandoraden. Bildens yttersta pixlar ska täckas med en ram med bredd och färg enligt kommandoradsargumenten.
   När du har löst uppgiften ska

   ```
   java E8 baboon.jpg 10 0 0 0
   ```

   resultera i



För att avsluta programmet kan du behöva göra Ctrl-C.

9. I filen *E9.java* finns ett Java program som du ska kompilera och köra med

   ```
   java E9 baboon.jpg 50 20 200 40
   ```

   Du bör inte se något hända: programmet gör inget mer än att läsa in kommandoradsargumenten. Dessa argument står för

   - namnet till en fil som innehåller en bild
   - en kolumn
   - en rad
   - en bredd
   - en höjd

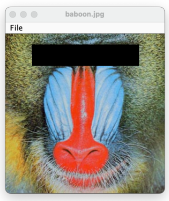
   Det du ska göra är att komplettera programmet (`main`) för att skapa ett objekt av typen `Picture` från bilden i filen med det namn som läses in från kommandoraden. På bilden ska en svart rektangel ritas och bilden ska visas i datorns skärm. Rektangeln ska

   - börja till vänster i den kolumn som anges i kommandoraden
   - börja från toppen i den rad som anges i kommandoraden
   - ha bredden och höjden som anges i kommandoraden

   När du har löst uppgiften ska

   ```
   java E9 baboon.jpg 50 20 200 40
   ```

   resultera i

   För att avsluta programmet kan du behöva göra Ctrl-C.

Inlämningsuppgifter - del 2

I filen *Bonus.java* finns ett java program som skapar ett objekt av typ `Picture` från en fil. Filnamnet anges i kommandoraden. Programmet skapar sedan två bilder: en oskarpare (`blurry`) och en skarpare (`sharp`). För att kunna göra det använder det två funktioner implementerade som metoder i samma fil. Du ska komplettera dessa metoder. Förklaringarna finns i kommentarer direkt ovanför metoderna. För att kunna använda programmet behöver du ha bibliotekklassen `NumericalArrays` i samma mapp.

Uppgifter från gamla tentor som kan lösas efter vecka 4.

1. Denna uppgift handlar om datatypen `Color` som vi använde i kursen. APIn för klassen `Color` finns i referenskortet som följer med denna tenta.
   Komplettera koden för att skapa en färg som har röd-grön-blå komponenter som är medelvärdet av motsvarande komponenter i färgerna `c1` och `c2`. Färgen ska tilldelas variabeln `c`.

   I koden tvingar vi divisionen att göras som flyttals division (genom att använda literalen `2.0`) för att sedan konvertera resultatet till heltal. Vi använder metoderna i klassen `Color` för att ta reda på komponenternas värden i färgerna `c1` och `c2`. Vi använder konstrueraren för att skapa en färg med de önskade komponenterna.

   ```
   Picture p = new Picture("photo.jpg");
   Color c1 = p.get(0,0);
   Color c2 = p.get(1,1);
   Color c;

   // Din kod här.
   // Avsluta med att tilldela färgen till variablen c.
   int red   = (int) ((c1.getRed()   + c2.getRed())   / 2.0);
   int green = (int) ((c1.getGreen() + c2.getGreen()) / 2.0);
   int blue  = (int) ((c1.getBlue()  + c2.getBlue())  / 2.0);
   c = new Color(red,green,blue);

   p.set(0,0,c);
   ```

2. Denna uppgift handlar om datatypen `Color` som vi använde i kursen. APIn för klassen `Color` finns i referenskortet som följer med denna tenta.
   Komplettera koden för att skriva ut röd-grön-blå komponenterna i färgerna `c1` och `c2`.

   I koden skriver vi ut alla tre komponenter för `c1` på en rad och alla komponenter för `c2` på nästa rad. Vi använder metoderna i klassen `Color` för att ta reda på komponenternas värden i färgerna `c1` och `c2`.

   ```
   Picture p = new Picture("photo.jpg");
   Color c1 = p.get(0,0);
   Color c2 = p.get(1,1);

   // Din kod här.
   System.out.println(c1.getRed() + " " + c1.getGreen() + " " + c1.getBlue());
   System.out.println(c2.getRed() + " " + c2.getGreen() + " " + c2.getBlue());
   ```

3. Denna uppgift handlar om datatypen `Picture` som vi använde i kursen. APIn för klassen `Picture` finns i referenskortet som följer med denna tenta.
   Komplettera koden för göra att pixlarna i positionerna högst upp till vänster och längst ner till höger i `p` byter färg med varandra.

   ```
   Picture p = new Picture("photo.jpg");

   // Din kod här.
   ```

   Lösningsförslag:

```
Picture p = new Picture("photo.jpg");
int w = p.width();
int h = p.height();
Color ctl = p.get(0,0);
Color cbr = p.get(w-1, h-1);
p.set(0,0,cbr);
p.set(w-1, h-1, ctl);
```

Positionen högst upp till vänster har koordinater (0, 0). Positionen längst ner till höger har koordinater (bredd-1, höjd -1).

4. Denna uppgift handlar om datatypen `Picture` som vi använde i kursen. APIn för klassen `Picture` finns i referenskortet som följer med denna tenta.
Komplettera koden för att göra att alla pixlar i p där x koordinaten är lika med y koordinaten får färgen svart.

```
Picture p = new Picture("photo.jpg");

// Din kod här.
```

Lösningsförslag:

```
Picture p = new Picture("photo.jpg");
int w = p.width();
int h = p.height();
for(int x = 0; x < w; x++){
    for(int y = 0; y < h; y++){
        if (x == y){
          p.set(x,y, Color.BLACK);
        }
    }
}
```