HÖGSKOLAN
I HALMSTAD

Verónica Gaspes
ITE
1 december 2022

Inlämningsuppgifter för vecka 6

Anmärkningar

- Hur man lämnar in och vad som ska lämnas in förklaras i Blackboard. Denna dokument beskriver uppgifterna. Uppgifterna finns i Java filer som refereras i varje uppgift i detta dokument. I varje .java fil finns en kommentar som förklarar uppgifterna på engelska.
- Uppgifterna 1 och 2 finns i filen *BouncingBall.java*. Uppgifterna 3, 4, 5, 6 finns i filen *ManyBalls.java*. Uppgifterna 7, 8, 9, 10, 11, 12 finns i filen *Complex.java*. Bonus uppgiften finns i filen *IntSet.java*.
- När du löser veckans uppgifter behöver du bara använda det vi har presenterat under första, andra, tredje, fjärde, femte och sjätte 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
  - att definiera abstrakta datatyper med klasser med instansvariabler, konstanter, konstruerare och icke statiska metoder.
- 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

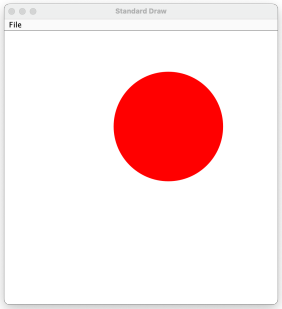
- använda standard input och omdirigering av standard input och standard output
- vad en statisk metod i Java är och hur de kan användas för att definiera funktioner och metoder i Java
- definiera och använda statiska metoder i ett program
- förklara vad en klass bibliotek är samt definiera statiska 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
- definiera klasser som implementerar en abstrakt datatyp

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. Filen *BouncingBall.java* innehåller definitionen av en abstrakt datatyp för studsbollar: klassen `BouncingBall`. Klassen har även även en metod `main` som fungerar som ett litet test som visar hur studsbollar kan användas. Du ska kompilera och köra programmet. Det är `main` som körs när du kör programmet så du ska läsa koden för `main` för att förstå det som händer. Det du bör se är en utskrift och ett StdDraw fönster enligt:

```
java BouncingBall
[Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, 0.15)]
[Ball at (0.6, 0.65) with radius 0.2 and velocity (0.1, 0.15)]
[Ball at (0.0, 0.0) with radius 0.0 and velocity (0.0, 0.0)]
[Ball at (0.0, 0.0) with radius 0.0 and velocity (0.0, -0.0)]
Exception in thread "main" java.lang.IllegalArgumentException: color is null
at StdDraw.validateNotNull(StdDraw.java:748)
at StdDraw.setPenColor(StdDraw.java:933)
at BouncingBall.draw(BouncingBall.java:44)
at BouncingBall.main(BouncingBall.java:95)
```



Det du ser i utskriften är text beräknad av metoden `toString` som anropas automatiskt i `System.out.println`. Du ser även ett exekveringsfel. Exekveringsfelet kommer från att i `main` skapas en studsboll med den konstruerare som inte är färdig programmerad.
Det du ska göra i denna uppgift är att komplettera koden för en konstruerare med signatur

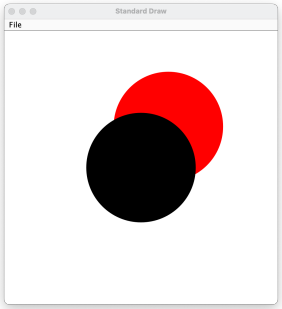
```
public BouncingBall(double cx, double cy,
                    double rad,
                    double velX, double velY)
```

Konstrueraren ska bygga upp en studsboll med mittpunkt i `(cx, cy)`, radie `rad` och rörelse steg `(velX, velY)`. Bollens färg ska vara svart. Du kan få inspiration från definitionen av den redan programmerade konstrueraren med signatur

```
public BouncingBall(double cx, double cy,
                    double rad,
                    Color col,
                    double velX, double velY)
```

Denna redan programmerade konstruerare tar även en färg som argument för att bestämma färgen på den boll som byggs upp.
När du är klar och kör programmet ska utskriften och fönstret bli

```
java BouncingBall
[Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, 0.15)]
[Ball at (0.6, 0.65) with radius 0.2 and velocity (0.1, 0.15)]
[Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, 0.15)]
[Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, -0.15)]
```



2. I samma fil ska du ändra metoden `bounceY()` med definitionen

   ```
   public void bounceY(){
     stepY = -stepY;
   }
   ```

   Metoden bestämmer vad som händer när bollen studsar i vertikal riktning: steget i rörelsen i y-led ändrar tecken!
   Det du ska göra är att ändra i metoden så att bollen även blir långsammare med 20 %: steget i rörelsen i y-led ändrar tecken och behåller bara 80 % av dess värde.
   När du är klar och kör programmet ska utskriften bli:

   ```
   java BouncingBall
   [Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, 0.15)]
   [Ball at (0.6, 0.65) with radius 0.2 and velocity (0.1, 0.15)]
   [Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, 0.15)]
   [Ball at (0.5, 0.5) with radius 0.2 and velocity (0.1, -0.12)]
   ```

3. Filen *`ManyBalls.java`* innehåller en klass med ett program (`main`) som läser in ett antal studsbollar från standard input och låter de röra sig i en rityta med koordinater från -10 till 10 i både x-led och y-led.

   I denna uppgift ska du programmera metoden `nextBall` med signaturen

   ```
   public static BouncingBall nextBall(Scanner data)
   ```

   Metoden ska använda argumentet `data` som är en `Scanner`, för att läsa in åtta tal med hjälp av `Scanners` metoder:

   - (1) x värdet för en bolls mittpunkt (en `double`)
   - (2) y värdet för en bolls mittpunkt (en `double`)
   - (3) värdet för en bolls radie (en `double`)
   - (4,5,6) värdena för RGB komponenterna i en bolls färg (tre `int`)
   - (7) värdet för en bolls steg i x-led (en `double`)

- (8) värdet för en bolls steg i y-led (en `double`)

Med dessa värden ska en boll skapas och returneras av metoden.
Du kan använda följande program för att testa din metod:

```
import java.util.Scanner;
import java.util.Locale;

public class TestNextBall{
    public static void main(String[] args){
        Scanner inputStream = new Scanner(System.in).useLocale(Locale.US);
        BouncingBall aBall =  ManyBalls.nextBall(inputStream);
        System.out.println(aBall);
    }
}
```

Programmet kan användas som i

```
java TestNextBall
1 2 3 4 5 6 7 8
```

då blir utskriften

```
[Ball at (1.0, 2.0) with radius 3.0 and velocity (7.0, 8.0)]
```

4. I samma fil ska du programmera metoden `readAll` med signaturen

   ```
   public static BouncingBall[] readAll(Scanner data, int n){
   ```

   Metoden ska använda argumentet `data` för att läsa in `n` bollar från standard input (`n` är andra argumentet till metoden). Varje boll ska läsas med metoden `nextBall`. Alla bollar ska läggas i ett fält (array) och det är detta fält som är resultaten av metoden.
   Du kan utöka testprogrammet för att använda denna metod och testa med att läsa in flera bollar från kommandoraden och skriva ut de till standard output.

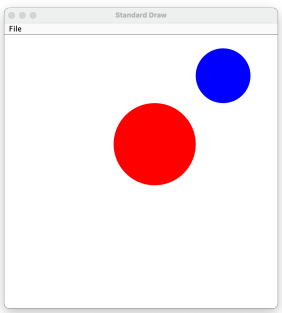
5. I samma fil ska du programmera metoden `drawAll` med signaturen

   ```
   public static void drawAll(BouncingBall[] balls)
   ```

   som ska rita alla bollar i argumentet `balls` i StdDraw (filen *`StdDraw.java`* finns med i zip-filen). Varje boll ska ritas med operationen `draw` från datatypen `BouncingBall`.
   Du kan nu testa programmet. När du kör

   ```
   java ManyBalls 2
   1 2 3 255 0 0 0 0
   6 7 2 0 0 255 0 0
   ```

   bör du se ett StdDraw fönster med



6. I samma fil ska du programmera metoden `moveAll` med signaturen

   ```
   public static void moveAll(BouncingBall[] balls, int bound)
   ```

   Första argumentet är ett fält med bollar. Metoden ska få alla bollar att göra ett steg av rörelse. Varje boll ska flyttas med operationen `move` från datatypen `BouncingBall`. Observera att bollen ska studsa om den når kanten. För att kolla om den når kanten ska du använda den andra argumentet: bollen finns i en kvadrat med gränser vid `-bound` och `bound` i både x- och y-led.
   Du kan testa programmet med

   ```
   java ManyBalls 2
   1 2 3 255 0 0 .1 .2
   6 7 2 0 0 255 .2 .1
   ```

   och du bör se att de två bollar rör sig och studsar mot kanterna.

   Du kan även testa programmet med

   ```
   java RandomBalls 30 10 | java ManyBalls 30
   ```

   Du bör då se ett StdDraw fönster med 30 små bollar med olika färger och olika hastigheter som rör sig och studsar mot kanterna. Programmet `RandomBalls` skriver till standard output 30 bollar, alla med samma radie men placerade på slumppositioner, med slumpfärger och slumphastigheter i en kvadratiska värld med gränser -10 10. Filen *`RandomBalls.java`* finns med i zip-filen.

7. Filen *`Complex.java`* innehåller definitionen av en abstrakt datatyp för komplexa tal: klassen `Complex`. Klassen har även även en metod `main` som fungerar som ett litet test som visar hur komplexa tal kan användas.
   Du ska kompilera och köra programmet. Det är `main` som körs när du kör programmet så du ska läsa koden för `main` att förstå det som händer.

   ```
   java Complex
   i*i = -1.0
   2.0 + 3.0i plus 5.0 + -4.0i = 7.0 + -1.0i
   2.0 + 3.0i scaled with -1 is: null
   2.0 + 3.0i minus 5.0 + -4.0i = null
   The absolute value of 2.0 + 3.0i is: 0.0
   The conjugate of 2.0 + 3.0i is: null
   5.0 + 15.0i quot 1.0 + -3.0i = null
   ```

   I denna uppgift ska du förbättra metoden `toString` så att den text som används för ett komplext tal med negativt imaginär del använder – tecknet för att bygga upp talet. Till exempel ska det komplexa talet med reell del 3.0 och imaginär del -4.0 beskrivas av texten $3.0 - 4.0i$ istället för $3.0 + -4.0i$. När du kör programmet ska utskriften bli (observera andra raden):

   ```
   i*i = -1.0
   2.0 + 3.0i plus 5.0 - 4.0i = 7.0 - 1.0i
   2.0 + 3.0i scaled with -1 is: null
   2.0 + 3.0i minus 5.0 + -4.0i = null
   The absolute value of 2.0 + 3.0i is: 0.0
   The conjugate of 2.0 + 3.0i is: null
   5.0 + 15.0i quot 1.0 + -3.0i = null
   ```

8. I samma fil ska du komplettera definitionen av metoden `scale` med signaturen

   ```
   public Complex scale(double x)
   ```

   Metoden ska multiplicera ett komplext tal med ett reellt tal. Detta beskrivs i Räkneregler för komplexa tal i kursen Algebra och diskret matematik. Här är formeln: $x(a+ib) = xa+ixb$. Du kan kolla implementationen av metoden `toString` för att se hur du kan använda det aktuella komplexa talet.
   När du kör programmet blir utskriften (observera tredje raden):

   ```
   i*i = -1.0
   2.0 + 3.0i plus 5.0 - 4.0i = 7.0 - 1.0i
   2.0 + 3.0i scaled with -1 is: -2.0 - 3.0i
   2.0 + 3.0i minus 5.0 + -4.0i = null
   The absolute value of 2.0 + 3.0i is: 0.0
   The conjugate of 2.0 + 3.0i is: null
   5.0 + 15.0i quot 1.0 + -3.0i = null
   ```

9. I samma fil ska du komplettera definitionen av metoden `minus` med signaturen

   ```
   public Complex minus(Complex that)
   ```

   Metoden ska subtrahera argumentet `that` från ett komplext tal. Hur man räknar minus mellan komplexa tal beskrivs i Räkneregler för komplexatal i kursen Algebra och diskret matematik. Formeln är $z_1 - z_2 = z_1 + (-1)z_2$. Du kan kolla implementationen av metoderna `plus` och `times` för att förstå hur man använder argumentet och det aktuella komplexa talet (`this`).
   När du kör programmet blir utskriften (observera fjärde raden):

   ```
   i*i = -1.0
   2.0 + 3.0i plus 5.0 - 4.0i = 7.0 - 1.0i
   2.0 + 3.0i scaled with -1 is: -2.0 - 3.0i
   2.0 + 3.0i minus 5.0 - 4.0i = -3.0 + 7.0i
   The absolute value of 2.0 + 3.0i is: 0.0
   The conjugate of 2.0 + 3.0i is: null
   5.0 + 15.0i quot 1.0 + -3.0i = null
   ```

10. I samma fil ska du komplettera definitionen av metoden `abs` med signaturen

    ```
    public double abs()
    ```

    för att beräkna det absoluta beloppet av ett komplext tal. Detta beskrivs i Absolutbelopp och konjugat i kursen Algebra och diskret matematik. Formeln är $|a+ib| = \sqrt{a^2+b^2}$.
    När du kör programmet blir utskriften (observera femte raden):

    ```
    i*i = -1.0
    2.0 + 3.0i plus 5.0 - 4.0i = 7.0 - 1.0i
    2.0 + 3.0i scaled with -1 is: -2.0 - 3.0i
    2.0 + 3.0i minus 5.0 - 4.0i = -3.0 + 7.0i
    The absolute value of 2.0 + 3.0i is: 3.605551275463989
    The conjugate of 2.0 + 3.0i is: null
    5.0 + 15.0i quot 1.0 + -3.0i = null
    ```

11. I samma fil ska du komplettera definitionen av metoden `conjugate` med signaturen

    ```
    public Complex conjugate()
    ```

    för att beräkna konjugatet av ett komplext tal. Detta beskrivs i Absolutbelopp och konjugat i kursen Algebra och diskret matematik. Formeln är $\overline{a+ib} = a - ib$.
    När du kör programmet blir utskriften (observera sjätte raden):

    ```
    i*i = -1.0
    2.0 + 3.0i plus 5.0 - 4.0i = 7.0 - 1.0i
    2.0 + 3.0i scaled with -1 is: -2.0 - 3.0i
    2.0 + 3.0i minus 5.0 - 4.0i = -3.0 + 7.0i
    The absolute value of 2.0 + 3.0i is: 3.605551275463989
    The conjugate of 2.0 + 3.0i is: 2.0 - 3.0i
    5.0 + 15.0i quot 1.0 + -3.0i = null
    ```

12. I samma fil ska du komplettera definitionen av metoden `quotient` med signaturen

    ```
    public Complex quotient(Complex that)
    ```

    för att beräkna kvoten av ett komplext tal med ett annat. Detta beskrivs i Division i kursen Algebra och diskret matematik. Formeln är $\frac{z_1}{z_2} = \frac{z_1\overline{z_2}}{|z_2|^2}$.
    När du kör programmet blir utskriften (observera sista raden):

    ```
    i*i = -1.0
    2.0 + 3.0i plus 5.0 - 4.0i = 7.0 - 1.0i
    2.0 + 3.0i scaled with -1 is: -2.0 - 3.0i
    2.0 + 3.0i minus 5.0 - 4.0i = -3.0 + 7.0i
    The absolute value of 2.0 + 3.0i is: 3.605551275463989
    The conjugate of 2.0 + 3.0i is: 2.0 - 3.0i
    5.0 + 15.0i quot 1.0 - 3.0i = -4.0 + 3.0i
    ```

Inlämningsuppgifter - del 2

I filen *IntSet.java* finns en klass som är tänkt att implementera en abstrakt datatyp där värdena är *mängd av (små) icke negativa heltal*. Med *små* menar man under en viss gräns. Här ser du några exempel av värden a typen mängd av heltal under 10000:

$$\{\}$$
$$\{0, 10, 1102\}$$
$$\{2, 5, 7, 11\}$$

Du ska komplettera implementationen enligt de anvisningar som finns i filen.

Uppgifter från gamla tentor som kan lösas efter vecka 4.

1. Denna uppgift handlar om dataypen `Balloon` med följande implementation:

```
public class Balloon{

    private  double centerX;
    private  double centerY;
    private  double rad;

    public Balloon(double x, double y, double r){
        centerX = x;
        centerY = y;
        rad = r;
    }

    public void draw(){
        StdDraw.filledCircle(centerX, centerY, rad);
    }
}
```

   (a) Lägg till en metod för att blåsa upp en ballong genom att öka ballongens radie med 10. Metoden ska ha signaturen

   ```
   public void blowUp()
   ```

   Metoden ändrar instansvariablen för radien genom att öka dess värde med 10:

   ```
   public void blowUp(){
      rad = rad + 10;
   }
   ```

   (b) Lägg till en metod för att få en ballong att flyga genom att öka ballongens centrums vertikala komponent. Metoden ska ha signaturen

   ```
   public void fly(double step)
   ```

   Metoden ändrar instansvariablen för mittpunktens y-koordinat genom att öka dess värde med argumentet `step`:

   ```
   public void fly(double step){
      centerY = centerY + step;
   }
   ```

2. Denna uppgift handlar om dataypen `BankAccount` med följande implementation:

```
public class BankAccount{
    private String personNumber;
    private int saldo;

    public BankAccount(String pn){
        personNumber = pn;
        saldo = 0;
    }
```

```
    public String toString(){
        return "Account belonging to " + personNumber + " with saldo: " + saldo;
    }

    public static void main(String[] args){
        BankAccount me = new BankAccount("111111-1111");
        me.deposit(1000);
        System.out.println(me);
    }
}
```

(a) Lägg till en metod för att kunna sätta in pengar i ett bankkonto så att saldot ökar:

`public void deposit(int value)`

Lösningsförslag:

```
public void deposit(int value){
   saldo = saldo + value;
}
```

Instansvariablen uppdateras genom att öka dess värde med `value`.

(b) Lägg till en metod för att kunna ta ut pengar från ett bank konto så att saldot minskar:

`public void retrieve(int value)`

Lösningsförslag:

```
public void deposit(int value){
   if(saldo >= value){saldo = saldo - value;}
}
```

Instansvariablen uppdateras genom att minska dess värde med `value` om det finns tillräcklig med pengar i kontot.