Laboration 14

Avsikten med laborationen är att ska skriva klasser som innehåller instansvariabler, instansmetoder och konstruktorer. Speciellt tränas att använda attribut som är referenser till andra objekt.

Skapa paketet **laboration14** innan du fortsätter med laborationen.

Grundläggande uppgifter

Uppgift 14a

Klassen ***PhoneNumber*** kan lagra ett antal telefonnummer (se klassdiagrammet till höger). Klassen innehåller tre instansvariabler:

- *home* (String) – telefonnummer hem
- *work* (String) – telefonnummer arbetet
- *mobile* (String) – mobilnummer

PhoneNumber
- home : String - work : String - mobile : String
+PhoneNumber(String, String, String) + setHome(String) + setWork(String) + setMobile(String) + getHome() : String + getWork() : String + getMobile() : String + toString() : String

Skriv klassen ***PhoneNumber*** i paketet *laboration14*.

Metoden ***toString*** ska returnera en sträng på formen:

```
Home:   040-123456
Work:   040-333333
Mobile: 0705223344
```

Om du exekverar nedanstående kod ska du få körresultatet nedanför koden.

```
PhoneNumber phone = new PhoneNumber( "0413-11111", "040-123456",
"0735191919" );
System.out.println( phone.getHome() + ", " + phone.getWork() + ", " +
                    phone.getMobile() );
System.out.println( "------------------------" );
System.out.println( phone.toString() );
System.out.println( "------------------------" );
phone.setHome("040-222222");
phone.setWork("");
phone.setMobile("0707657654");
System.out.println( phone.toString() );
```

Körresultat

```
0413-11111, 040-123456, 0735191919
------------------------
Home:   0413-11111
Work:   040-123456
Mobile: 0735191919
------------------------
Home:   040-222222
Work:
Mobile: 0707657654
```

Uppgift 14b

Skriv klassen ***Contact*** vilken ska representera en person i en telefonbok. Klassen innehåller två attribut, namnet på personen och ett *PhoneNumber*-objekt med personens olika telefonnummer.

- ***Konstruktorn*** ska ta emot fyra argument – namn, hemnummer, arbetsnummer och mobilnummer. Samtliga ska vara av typen String.
- ***toString***-metoden ska returnera ett String-objekt på formen:

```
Contact: Eva Bok
Home:   0413-11111
Work:   040-123456
Mobile: 0735191919
```

Contact
- name : String - phone : PhoneNumber
+ Contact(String, String, String, String) + getName() : String + getHome() : String + getMobile() : String + toString() : String

Om du exekverar nedanstående kod ska du få körresultatet nedanför koden.

```
Contact contact = new Contact("Eva Bok","040-123456","040-
333333","0705223344");
System.out.println( "Namn: " + contact.getName() + "\n" +
                    "Telefonnummer: " + contact.getHome() + ", " +
                    contact.getWork() + ", " + contact.getMobile() );
System.out.println("--------------------------------");
System.out.println( contact.toString() );
```

Körresultat

```
Namn: Eva Bok
Telefonnummer: 040-123456, 040-333333, 0705223344
--------------------------------
Contact: Eva Bok
Home:   040-123456
Work:   040-333333
Mobile: 0705223344
```

Uppgift 14c

Lägg till metoderna ***setName***, ***setHome***, ***setWork*** och ***setMobile*** i klassen ***Contact***. Testa deras funktion genom att lägga till följande instruktioner efter koden i *Uppgift 14b*.

Contact
- name : String - phone : PhoneNumber
: + setName(String) + setHome(String) + setWork(String) + setMobile(String) + toString() : String

```
System.out.println("--------------------------------");
contact.setHome("040-999999");
contact.setWork("0415-22222");
contact.setMobile("0776192837");
System.out.println( contact.toString() );
```

Körresultat

```
Namn: Eva Bok
Telefonnummer: 040-123456, 040-333333, 0705223344
--------------------------------
Contact: Eva Bok
Home:   040-123456
Work:   040-333333
Mobile: 0705223344
--------------------------------
Contact: Eva Bok
Home:
Work:   0415-22222
Mobile: 0776192837
```

Uppgift 14d

Klasserna **Building** och **Address** är givna (se klassdiagram till höger). Klassen **Building** representerar en byggnad och innehåller instansvariablerna *floors* (våningar) och *area* (bostadsyta). Klassen **Address** representerar en adress och innehåller instansvariablerna *street* (gata), *postalCode* (postadress) och *town* (ort).

Placera klasserna i paketet *laboration14* och testkör main-metoden i vardera klassen.

Building

- floors : int
- area : int

+ Building(int, int)

+ getFloors() : int
+ getArea() : int
+ toString() : String

Address

- street : String
- postalCode : int
- town : String

+Address(String, int, String)
+ getStreet() : String
+ getPostalCode() : int
+ getTown() : String
+ toString() : String

RealEstate

- name : String
- building : Building
- address : Address

+RealEstate(String, Building, Address)
+ getName() : String
+ getBuilding() : Building
+ getAddress() : Address
+ toString() : String

Skriv klassen ***RealEstate*** (fastighet, se klassdiagram till höger) vilken innehåller attribut för

- fastighetens beteckning (name),
- byggnaden (building)
- fastighetens adress (address)

Följande gäller för klassen:

- Klassen ska innehålla en ***konstruktor*** som tar argument för samtliga attribut (instansvariabler).
  public RealEstate(String, Building, Address)
- Klassen ska innehålla ***get***-metoder till attributen.
- Klassen ska innehålla metoden ***toString*** vilken ska returnera ett String-objekt på formen:

```
Fastighet: Bryggeriet 6
Byggnad: 3 våningar, yta 343 kvm
Adress: Algatan 4, 26733 Malmö
```

  toString-metoderna i klasserna *Building* och *Address* ska du använda i din lösning (testa dem – de returnerar lämpliga String-objekt).

Din lösning ska kunna köras med programmet:

```
Building building = new Building(5, 1350);
Address address = new Address("Nygatan 4", 12345,
"Ystad");
RealEstate re = new RealEstate("Kvarnen 4", building, address);
System.out.println(re.getName());
System.out.println(re.getBuilding().toString()); // toString i Building-
klassen
System.out.println(re.getAddress().toString());  // toString i Address-
klassen
System.out.println("--------------------------");
System.out.println(re.toString());
```

Körresultat

```
Kvarnen 4
Byggnad: 5 våningar, yta 1350 kvm
Adress: Nygatan 4, 12345 Ystad
--------------------------
Fastighet: Kvarnen 4
Byggnad: 5 våningar, yta 1350 kvm
Adress: Nygatan 4, 12345 Ystad
```

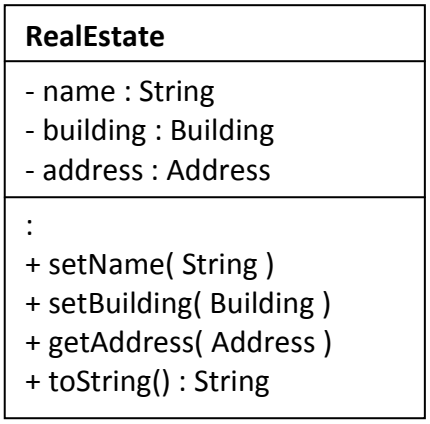
Uppgift 14e

Lägg till metoderna ***setName***, ***setBuilding*** och ***setAddress*** i klassen ***RealEstate***. Testa deras funktion genom att lägga till följande instruktioner efter koden i *Uppgift 14d*.

```
System.out.println("--------------------------");
re.setName("Skorpionen 17");
re.setBuilding( new Building(6, 1470) );
re.setAddress( new Address("Nygatan 4", 55555, "Ystad") );
System.out.println(re.toString());
```

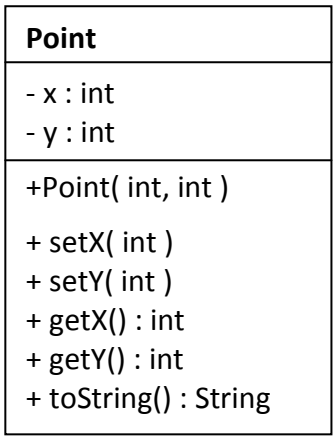
Körresultat

```
Kvarnen 4
Byggnad: 5 våningar, yta 1350 kvm
Adress: Nygatan 4, 12345 Ystad
--------------------------
Fastighet: Kvarnen 4
Byggnad: 5 våningar, yta 1350 kvm
Adress: Nygatan 4, 12345 Ystad
--------------------------
Fastighet: Skorpionen 17
Byggnad: 6 våningar, yta 1470 kvm
Adress: Nygatan 4, 55555 Ystad
```



Uppgift 14f

Gör ett **klassdiagram** som beskriver klassen **Circle** nedan. Klassen **Point** är given och beskrivs av klassdiagrammet till höger.




```
public class Circle {
    private double radius;
    private Point position;

    public Circle( double inRadius, Point inPosition ) {
        this.radius = inRadius;
        this.position = inPosition;
    }

    public void setPosition( Point inPosition ) {
        this.position = inPosition;
    }

    public double getRadius() {
        return this.radius;
    }

    public int getX() {
        return this.position.getX();
    }

    public int getY() {
        return this.position.getY();
    }

    public Point getPosition() {
        return this.position;
    }

    public String toString() {
        return "CIRCLE, radius = " + this.radius + ", position = " +
position.toString();
    }
}
```

Uppgift 14g

Placera klasserna ***Circle*** och ***Point*** i paketet *laboration14*. Kör nedanstående kod och studera körresultatet. Hur kan en ändring av innehållet i *p2* påverka cirkelns position?

```
Point p1, p2;
p1 = new Point( 10, 12 );
Circle c1 = new Circle( 3.5, p1 );
System.out.println( "p1 = " + p1.toString() );
System.out.println( "c1 = " + c1.toString() );
System.out.println("------------------------");
p2 = c1.getPosition();
p2.setX( 333 ); // x-värdet i p2 ändras till 333
System.out.println( "p1 = " + p1.toString() );
System.out.println( "c1 = " + c1.toString() );
System.out.println( "p2 = " + p1.toString() );
```

Körresultat

```
p1 = (10,12)
c1 = CIRCLE, radius = 3.5, position = (10,12)
------------------------
p1 = (333,12)
c1 = CIRCLE, radius = 3.5, position = (333,12)
p2 = (333,12)
```

Hur kan en ändring av x-värdet i *p2* påverka

a. värdet på x i Point-objektet *p1*?

b. cirkelns position?

Fördjupande uppgifter

Uppgift 14h

Lägg till konstruktorn

public Point(Point point)

i klassen ***Point***. Konstruktorn ska initiera instansvariablerna *x* resp *y* med motsvarande värde i argumentet *point*. Det nya Point-objektet ska alltså ha samma tillstånd som argumentet *point*.

Om du exekverar koden:

```
Point p1 = new Point( 10, 12 );
Point p2 = new Point( p1 );
System.out.println( p1.toString() );
System.out.println( p2.toString() );
```

ska du få körresultatet till höger.

```
(10,12)
(10,12)
```

Uppgift 14i

Lägg till metoden

public Point copy()

i klassen ***Point***. Metoden ska skapa ett Point som har samma tillstånd (värde i instansvariablerna) som aktuellt Point-objekt.

Om du exekverar koden:

```
Point p1 = new Point( 11, 13 );
Point p2 = p1.copy();  // Det nya Point-objektet ska ha tillståndet x=11 och y=13 (samma som p1)
System.out.println( p1.toString() );
System.out.println( p2.toString() );
```

ska du få körresultatet till höger.

```
(11,13)
(11,13)
```

Uppgift 14j

Lägg till metoden

public void setPoint(Point point)

i klassen ***Point***. Metoden ska tilldela instansvariablerna *x* resp *y* med motsvarande värde i argumentet *point*.

Om du exekverar koden:

```
Point p1 = new Point( 10, 12 );
Point p2 = new Point( 20, 15 );
p1.setPoint( p2 );
System.out.println( p1.toString() );
System.out.println( p2.toString() );
```

ska du få körresultatet till höger.

```
(20,15)
(20,15)
```

Uppgift 14k

Ändra i metoden

public Point getPosition()

i klassen ***Circle*** så att metoden returnerar en *kopia* av instansvariabeln *position*. Använd *copy*-metoden i klassen Point.

Om du exekverar koden i uppgift 14g så ska körresultatet bli:

Körresultat

```
p1 = (10,12)
c1 = CIRCLE, radius = 3.5, position = (10,12)
------------------------
p1 = (10,12)
c1 = CIRCLE, radius = 3.5, position = (10,12)
p2 = (10,12)
```

Ändringen av x-värdet i *p2* påverkar inte x-värdet i *p1* eller cirkelns position längre. Hur kommer det sig?

Uppgift 14l

Om du kör nedanstående kod så ändras Circle-objektets (*c1*:s) position när x-värdet i *p1* förändras. Detta beror ju på att *c1* använder samma Point-objekt som *p1* refererar till.

```
Point p1;
p1 = new Point( 10, 12 );
Circle c1 = new Circle( 3.5, p1 );
System.out.println( "p1 = " + p1.toString() );
System.out.println( "c1 = " + c1.toString() );
System.out.println("------------------------");
p1.setX( 444 );
System.out.println( "p1 = " + p1.toString() );
System.out.println( "c1 = " + c1.toString() );
```

Körresultat

```
p1 = (10,12)
c1 = CIRCLE, radius = 3.5, position = (10,12)
------------------------
p1 = (444,12)
c1 = CIRCLE, radius = 3.5, position = (444,12)
```

För att åtgärda detta ska du se till att Circle-objektet alltid har ett eget Point-objekt, dvs. ett Point-objekt som det inte finns några externa referenser till.

Gör följande:

- Ändra konstruktorn *Circle(double, Point)* så att en kopia av Point-argumentet lagras i instansvariabeln *position*. Kopian finns det inga referenser till utom den i instansvariabeln *position*. En kopia får du genom att skapa ett nytt Point-objekt med vettiga argument / genom att anropa copy-metoden.
- Ändra metoden *setPosition(Point)* så att *Point*-objektet som *position* refererar till får samma x- resp y-värde som Point-argumentet.

Efter ovanstående ändringar ger ovanstående kod följande körresultat.

Körresultat

```
p1 = (10,12)
c1 = CIRCLE, radius = 3.5, position = (10,12)
------------------------
p1 = (444,12)
c1 = CIRCLE, radius = 3.5, position = (10,12)
```

Lösningsförslag

Uppgift 14a

```
public class PhoneNumber {
    private String home;
    private String work;
    private String mobile;

    public PhoneNumber(String home, String work, String mobile) {
        super();
        this.home = home;
        this.work = work;
        this.mobile = mobile;
    }

    public void setHome(String home) {
        this.home = home;
    }

    public void setWork(String work) {
        this.work = work;
    }

    public void setMobile(String mobile) {
        this.mobile = mobile;
    }

    public String getHome() {
        return home;
    }

    public String getWork() {
        return work;
    }

    public String getMobile() {
        return mobile;
    }

    public String toString() {
        return "Home:   " + this.home + "\n" +
               "Work:   " + this.work + "\n" +
               "Mobile: " + this.mobile;
    }
}
```

Uppgift 14b

```
public class Contact {
    private String name;
    private PhoneNumber phone;

    public Contact(String name, String home, String work, String mobile) {
        this.name = name;
        this.phone = new PhoneNumber(home,work,mobile);
    }

    public String getName() {
        return this.name;
    }

    public String getHome() {
        return this.phone.getHome();
```

```
    }

    public String getWork() {
        return this.phone.getWork();
    }

    public String getMobile() {
        return this.phone.getMobile();
    }

    public String toString() {
        return "Contact: " + name + "\n" + this.phone.toString();
    }
}
```

Uppgift 14c

```
public class Contact {
    :
    public void setName(String name) {
        this.name = name;
    }

    public void setHome(String home) {
        this.phone.setHome(home);
    }

    public void setWork(String work) {
        this.phone.setWork(work);
    }

    public void setMobile(String mobile) {
        this.phone.setMobile(mobile);
    }
    :
}
```

Uppgift 14d

```
public class RealEstate {
    private String name;
    private Building building;
    private Address address;

    public RealEstate(String name, Building building, Address address) {
        this.name = name;
        this.building = building;
        this.address = address;
    }

    public String getName() {
        return this.name;
    }

    public Building getBuilding() {
        return this.building;
    }

    public Address getAddress() {
        return this.address;
    }

    public String toString() {
        return "Fastighet: " + this.name + "\n" +
```

```
                this.building.toString() + "\n" +
                this.address.toString();
    }
}
```

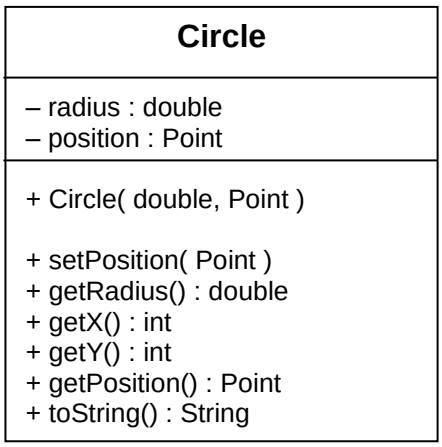
Uppgift 14e

```
public class RealEstate {
    :
    public void setName(String name) {
        this.name = name;
    }

    public void setBuilding(Building building) {
        this.building = building;
    }

    public void setAddress(Address address) {
        this.address = address;
    }
    :
}
```

Uppgift 14f



Uppgift 14g

När Circle-objektet skapas används variabeln *p1* som argument. Referensen i *p1* kopieras till parametern *inPosition*. Med instruktionen:

```
this.position = inPosition;
```

kopieras sedan samma referens till instansvariabeln *position*. Därför refererar *p1* och instansvarariabeln *position* till samma Point-objekt.
Metoden getPosition returnerar referensen som lagras i *position*. Denna referens tilldelas nu variabeln *p2*. Nu refererar *p1*, *p2* och instansvariabeln *position* till samma Point-objekt.

Uppgift 14h + i + j

```
public class Point {
    :
    // h
    public Point( Point point ) {
        this.x = point.getX();
        this.y = point.getY();
    }

    // i
```

```
    public Point copy() {
        Point point = new Point( this.x, this.y);
        return point;
    }

    // j
    public void setPoint( Point point ) {
        this.x = point.getX();
        this.y = point.getY();
    }
    :
}
```

Uppgift 14k + l

```
public class Circle {
    private double radius;
    private Point position;

    // l
    public Circle( double inRadius, Point inPosition ) {
        this.radius = inRadius;
        this.position = new Point(inPosition.getX(),inPosition.getY());
    }

    // k
    public Point getPosition() {
        return this.position.copy();
    }

    // l
    public void setPosition( Point inPosition ) {
        this.position.setX( inPosition.getX() );
        this.position.setY( inPosition.getY() );
    }
    :
}
```