Politechnika Wrocławska

Projektowanie Obiektowe 3

dr inż. Paweł Trajdos

Politechnika Wrocławska, Katedra Systemów i Sieci Komputerowych
Wyb. Wyspianskiego 27, 50-370 Wroclaw

5 lutego 2023

Politechnika Wrocławska

Spis treści

Politechnika Wrocławska

Section 1

Geometria wielowymiarowa – kontynuacja

Politechnika Wrocławska

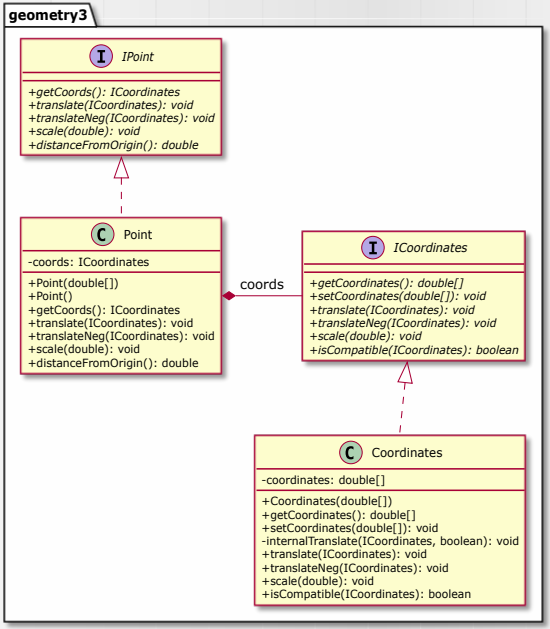
Analiza czasownikowo - rzeczownikowa

Potrzebny jest obiekt umożliwiający wykonywanie operacji na punktach w przestrzeni wielowymiarowej. Dla każdego z punktu potrzebna jest operacja pobrania jego współrzędnych, translacja punktu o zadany wektor, skalowanie współrzędnych oraz obliczenie odległości punktu od środka układu współrzędnych.

Politechnika Wrocławska

Analiza czasownikowo - rzeczownikowa

Potrzebny jest obiekt umożliwiający wykonywanie operacji na punktach w przestrzeni wielowymiarowej. Dla każdego z punktu potrzebna jest operacja pobrania jego współrzędnych, translacja punktu o zadany wektor, skalowanie współrzędnych oraz obliczenie odległości punktu od środka układu współrzędnych.


geometry3
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
Point
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean

Politechnika Wrocławska

Co można ulepszyć?

Politechnika Wrocławska

Subsection 1

Metoda Wytwórcza

Politechnika Wrocławska

Creator

Przypisz klasie B odpowiedzialność tworzenia obiektów klasy A, gdy:

- klasa B zwiera/agreguje obiekty klasy A;
- klasa B 'zapisuje/rejestruje życie' instancji klasy A;
- klasa B blisko współpracuje z A;
- klasa B ma dane inicjalizacyjne potrzebne przy tworzeniu obiektu klasy A.

Politechnika Wrocławska

Pure Fabrication

Jak przydzielić odpowiedzialność by nie naruszyć zasad High Cohesion i Low Coupling a nie odpowiada nam rozwiązanie sugerowane przez Information Expert?

Przypisz zakres odpowiedzialności sztucznej lub pomocniczej klasie, która nie reprezentuje żadnego problemu domenowego. Nie narusz zasad High Cohesion i Low Coupling.

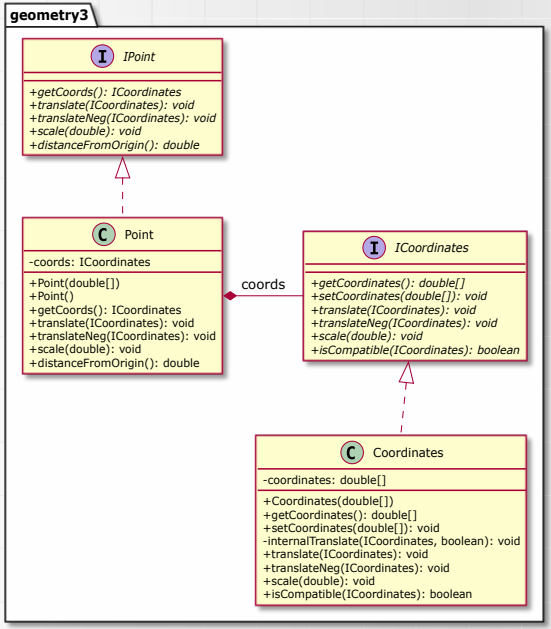
Politechnika Wrocławska

Zmiany

- Dependency Inversion Principle
- Open – Close Principle

Listing: Point.java

```
1  package geometry3;
2
3  public class Point implements IPoint {
4
5    private ICoordinates coords;
6
7    public Point(double[] coords) {
8      this.coords = new Coordinates(coords);
9    }
10   public Point(){this(new double[] {0,0});}
11
12   public ICoordinates getCoords() {
13     return  new Coordinates(coords.getCoordinates());
14   }
```


geometry3
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
Point
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean

Implementacja

Listing: Point.java

```
1  package geometry8;
2
3  public abstract class Point implements IPoint {
4    private ICoordinates coords;
5
6    public Point(double[] coords) {
7      this.coords = this.produceCoordinates(this.coords.getCoordinates());
8    }
9    public Point(){this(new double[] {0,0});}
10   public ICoordinates getCoords() {
11     return  this.produceCoordinates(this.coords.getCoordinates());
12   }
13
14   protected abstract ICoordinates produceCoordinates(double[] coord);
15
```

Implementacja

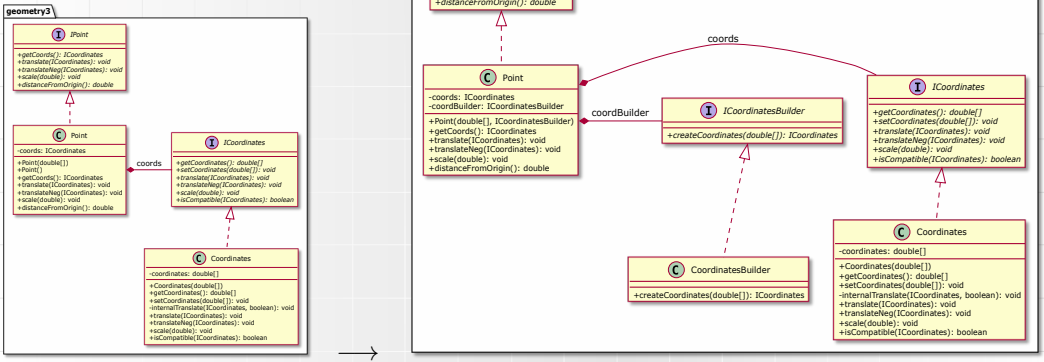
Listing: PointWithSimpleCoords.java

```java
package geometry8;

public class PointWithSimpleCoords extends Point {
  @Override
  protected ICoordinates produceCoordinates(double[] coord) {
    return new Coordinates(coord);
  }

}
```

Politechnika Wrocławska

Subsection 2

Budowniczy


geometry3
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
Point
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
geometry4
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
Point
-coords: ICoordinates
-coordBuilder: ICoordinatesBuilder
+Point(double[], ICoordinatesBuilder)
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coordBuilder
ICoordinatesBuilder
+createCoordinates(double[]): ICoordinates
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
CoordinatesBuilder
+createCoordinates(double[]): ICoordinates
Coordinates
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean

Implementacja

Listing: Point.java

```
1  package geometry4;
2
3  public class Point implements IPoint {
4    private ICoordinates coords;
5    private ICoordinatesBuilder coordBuilder;
6
7    public Point(double[] coords, ICoordinatesBuilder coordBuilder) {
8      this.coordBuilder = coordBuilder;
9      this.coords = coordBuilder.createCoordinates(coords);
10   }
11
12   public ICoordinates getCoords() {
13     return coordBuilder.createCoordinates(coords.getCoordinates());
14   }
15
```

Implementacja

Listing: ICoordinatesBuilder.java

```
1 package geometry4;
2
3 public interface ICoordinatesBuilder {
4
5   public ICoordinates createCoordinates(double[] coords);
6
7 }
```

Implementacja

Listing: CoordinatesBuilder.java

```java
package geometry4;

public class CoordinatesBuilder implements ICoordinatesBuilder {

  @Override
  public ICoordinates createCoordinates(double[] coords) {
    return new Coordinates(coords);
  }

}
```

Test

Listing: CoordinatesBuilderTest.java

```
1  package geometry4;
2
3  import static org.junit.Assert.assertTrue;
4
5  import org.junit.Test;
6
7  public class CoordinatesBuilderTest {
8
9    @Test
10   public void test() {
11     ICoordinatesBuilder builder = new CoordinatesBuilder();
12     assertTrue("Proper class", builder.createCoordinates(new double[] {1,2}) instanceof
        ICoordinates);
13   }
14
15 }
```

Politechnika Wrocławska

Subsection 3

Prototyp

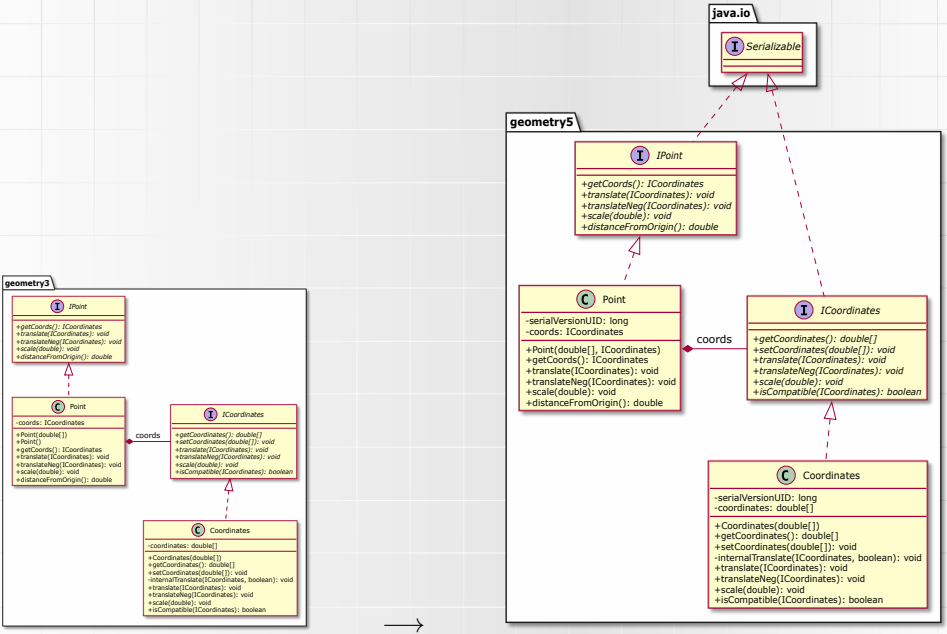
Politechnika Wrocławska

Zmiany

- Dependency Inversion Principle
- Open – Close Principle

Listing: Point.java

```
1  package geometry3;
2
3  public class Point implements IPoint {
4
5    private ICoordinates coords;
6
7    public Point(double[] coords) {
8      this.coords = new Coordinates(coords);
9    }
10   public Point(){this(new double[] {0,0});}
11
12   public ICoordinates getCoords() {
13     return  new Coordinates(coords.getCoordinates());
14   }
```


geometry3
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
Point
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
java.io
Serializable
geometry5
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
Point
-serialVersionUID: long
-coords: ICoordinates
+Point(double[], ICoordinates)
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-serialVersionUID: long
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean

Implementacja

Listing: Point.java

```
1  package geometry5;
2
3  import org.apache.commons.lang3.SerializationUtils;
4
5  public class Point implements IPoint {
6    private static final long serialVersionUID = 3764677738393141158L;
7    private ICoordinates coords;
8
9    public Point(double[] coords,ICoordinates cordsProto) {
10     this.coords = SerializationUtils.clone(cordsProto);
11     this.coords.setCoordinates(coords);
12   }
13
14   public ICoordinates getCoords() {
15     return  SerializationUtils.clone(coords);
16   }
```

Politechnika Wrocławska

Section 2

Geometria – Nowe wymagania

Analiza czasownikowo - rzeczownikowa

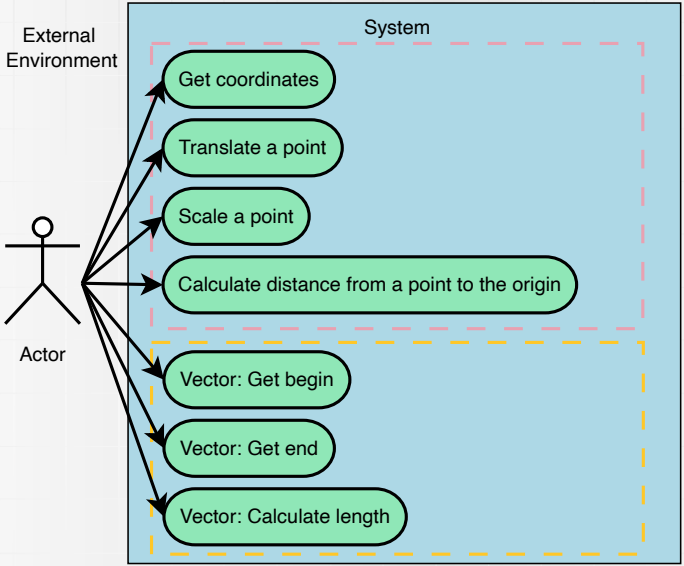
Potrzebny jest obiekt umożliwiający wykonywanie operacji na punktach w przestrzeni wielowymiarowej. Dla każdego z punktu potrzebna jest operacja pobrania jego współrzędnych, translacja punktu o zadany wektor, skalowanie współrzędnychs oraz obliczenie odległości punktu od środka układu współrzędnych.
Potrzebna będzie też reprezentacja wektora. Dla wektora potrzebne będą operacje pobrania punktów początkowego i końcowego oraz obliczenia jego długości.

Politechnika Wrocławska

Analiza czasownikowo - rzeczownikowa

Potrzebny jest obiekt umożliwiający wykonywanie operacji na punktach w przestrzeni wielowymiarowej. Dla każdego z punktu potrzebna jest operacja pobrania jego współrzędnych, translacja punktu o zadany wektor, skalowanie współrzędnychs oraz obliczenie odległości punktu od środka układu współrzędnych.
Potrzebna będzie też reprezentacja wektora. Dla wektora potrzebne będą operacje pobrania punktów początkowego i końcowego oraz obliczenia jego długości.

Diagram Przypadków użycia



Karty CRC

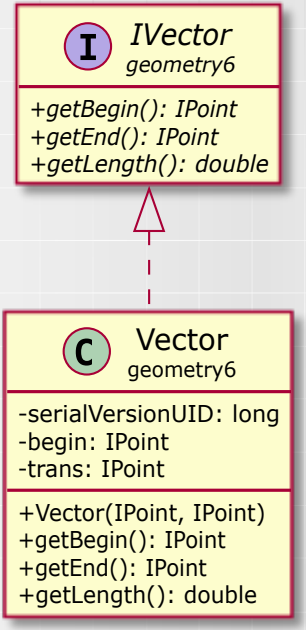
Classname: IVector

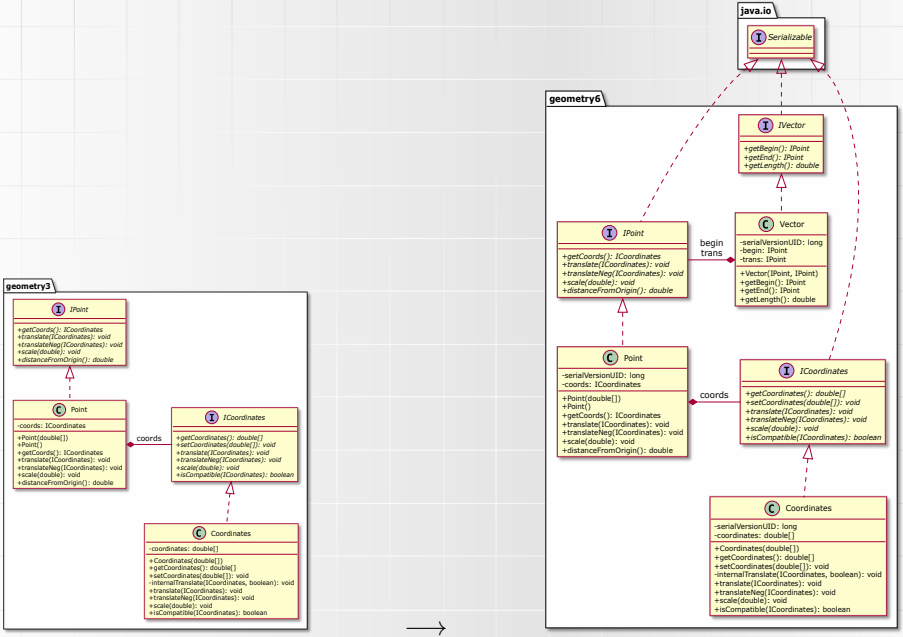
Superclass: none
Subclass(es): Vector

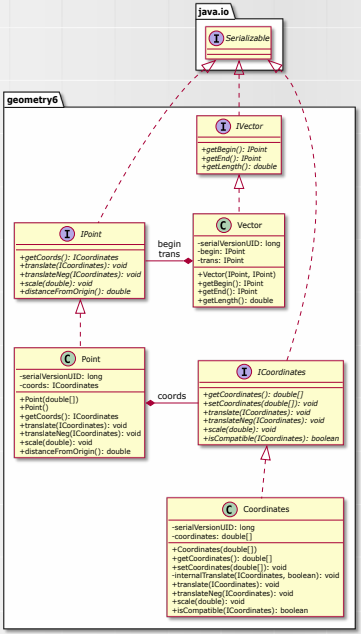
Responisibilities:
Handle Vector specific operations.

- Get begin
- Get end
- Calculate length

Collaboration:
IPoint


IVector
geometry6
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
Vector
geometry6
-serialVersionUID: long
-begin: IPoint
-trans: IPoint
+Vector(IPoint, IPoint)
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double


geometry3
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
Point
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
java.io
Serializable
geometry6
IVector
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
begin
trans
Vector
-serialVersionUID: long
-begin: IPoint
-trans: IPoint
+Vector(IPoint, IPoint)
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
Point
-serialVersionUID: long
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-serialVersionUID: long
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean


java.io
Serializable
geometry6
IVector
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
begin
trans
Vector
-serialVersionUID: long
-begin: IPoint
-trans: IPoint
+Vector(IPoint, IPoint)
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
Point
-serialVersionUID: long
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-serialVersionUID: long
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean

Implementacja

Listing: IVector.java

```
1  package geometry6;
2
3  import java.io.Serializable;
4
5  public interface IVector extends Serializable {
6
7    public IPoint getBegin();
8    public IPoint getEnd();
9    public double getLength();
10
11
12 }
```

Implementacja

Listing: Vector.java

```java
package geometry6;
import java.security.InvalidParameterException;

import org.apache.commons.lang3.SerializationUtils;
public class Vector implements IVector{
  private static final long serialVersionUID = -8316267110426165901L;
  private IPoint begin;
  private IPoint trans;

  public Vector(IPoint begin, IPoint trans) {
    if(!begin.getCoords().isCompatible(trans.getCoords()))
      throw new InvalidParameterException("Begin point and translation are incompatible");
    this.begin = SerializationUtils.clone(begin);
    this.trans = SerializationUtils.clone(trans) ;
  }
```

Implementacja

Listing: Vector.java

```
17    @Override
18    public IPoint getBegin() { return SerializationUtils.clone(begin); }
19
20    @Override
21    public IPoint getEnd() {
22      IPoint tmpPoint = SerializationUtils.clone(begin);
23      try {
24        tmpPoint.translate(trans.getCoords());
25      } catch (Exception e) {e.printStackTrace();}
26      return tmpPoint;
27    }
28
29    @Override
30    public double getLength() { return trans.distanceFromOrigin(); }
31  }
```

Testy

Listing: VectorTest.java

```
1  package geometry6;
2  import static org.junit.Assert.assertTrue;
3  import static org.junit.Assert.fail;
4
5  import org.junit.Test;
6
7  import complex5.DoubleComparator;
8
9  public class VectorTest {
10
11   @Test
12   public void test() {
13     IVector v1;
14     try {
15      v1 = new Vector(new Point(new double[] {1,2,3 }),new Point(new double[] {1,3}));
16     fail("Wrong constructor arguments -- no Exception");
17     }catch(Exception e) {
18       assertTrue("Wrong constructor arguments -- exception catched", true);
19     }
```

Testy

Listing: VectorTest.java

```
21      IPoint p1 = new Point(new double[] {0,0});
22      IPoint p2 = new Point(new double[] {1,1});
23      v1  = new Vector(p1, p2);
24      DoubleComparator dc = new DoubleComparator();
25      assertTrue("Length test", dc.eq(v1.getLength(), Math.sqrt(2.0)));
26      assertTrue("Get Begin",v1.getBegin().getCoords().equals(p1.getCoords()));
27      assertTrue("Get End", v1.getEnd().getCoords().equals(p2.getCoords()));
28    }
29  }
```

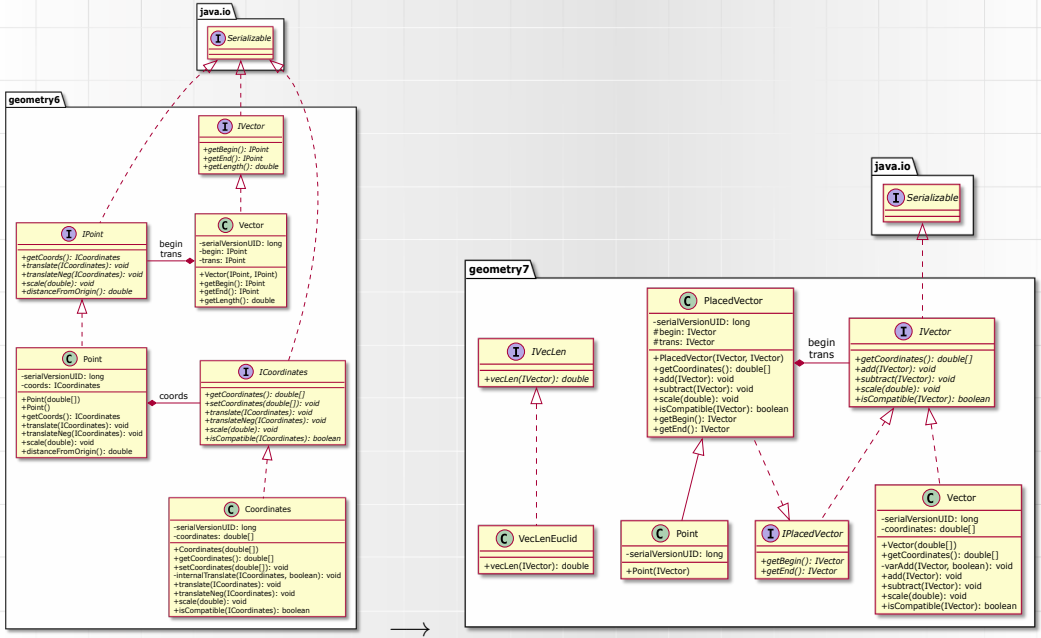
Politechnika Wrocławska

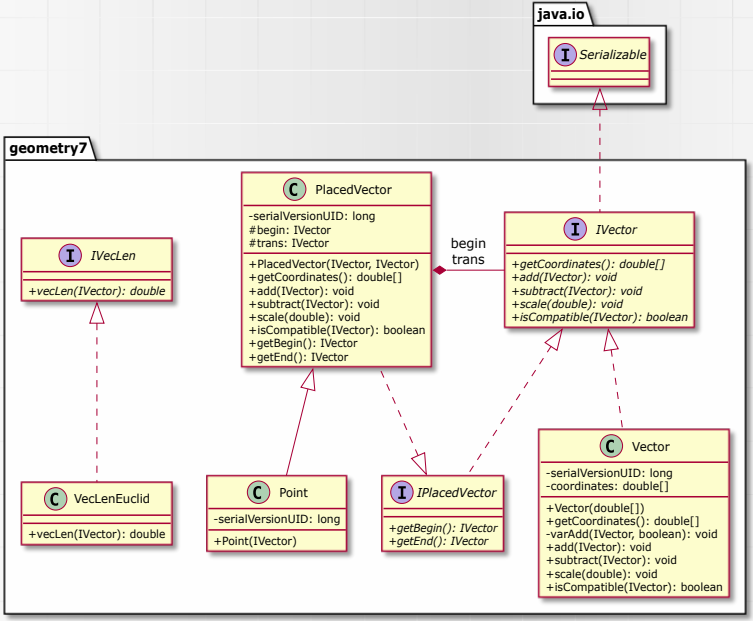
KISS!

Keep it simple, stupid!

+

SRP


java.io
Serializable
geometry6
IVector
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
IPoint
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
begin
trans
Vector
-serialVersionUID: long
-begin: IPoint
-trans: IPoint
+Vector(IPoint, IPoint)
+getBegin(): IPoint
+getEnd(): IPoint
+getLength(): double
Point
-serialVersionUID: long
-coords: ICoordinates
+Point(double[])
+Point()
+getCoords(): ICoordinates
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+distanceFromOrigin(): double
coords
ICoordinates
+getCoordinates(): double[]
+setCoordinates(double[]): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
Coordinates
-serialVersionUID: long
-coordinates: double[]
+Coordinates(double[])
+getCoordinates(): double[]
+setCoordinates(double[]): void
-internalTranslate(ICoordinates, boolean): void
+translate(ICoordinates): void
+translateNeg(ICoordinates): void
+scale(double): void
+isCompatible(ICoordinates): boolean
java.io
Serializable
geometry7
IVecLen
+vecLen(IVector): double
PlacedVector
-serialVersionUID: long
#begin: IVector
#trans: IVector
+PlacedVector(IVector, IVector)
+getCoordinates(): double[]
+add(IVector): void
+subtract(IVector): void
+scale(double): void
+isCompatible(IVector): boolean
+getBegin(): IVector
+getEnd(): IVector
begin
trans
IVector
+getCoordinates(): double[]
+add(IVector): void
+subtract(IVector): void
+scale(double): void
+isCompatible(IVector): boolean
VecLenEuclid
+vecLen(IVector): double
Point
-serialVersionUID: long
+Point(IVector)
IPlacedVector
+getBegin(): IVector
+getEnd(): IVector
Vector
-serialVersionUID: long
-coordinates: double[]
+Vector(double[])
+getCoordinates(): double[]
-varAdd(IVector, boolean): void
+add(IVector): void
+subtract(IVector): void
+scale(double): void
+isCompatible(IVector): boolean


java.io
Serializable
geometry7
PlacedVector
-serialVersionUID: long
#begin: IVector
#trans: IVector
+PlacedVector(IVector, IVector)
+getCoordinates(): double[]
+add(IVector): void
+subtract(IVector): void
+scale(double): void
+isCompatible(IVector): boolean
+getBegin(): IVector
+getEnd(): IVector
begin
trans
IVector
+getCoordinates(): double[]
+add(IVector): void
+subtract(IVector): void
+scale(double): void
+isCompatible(IVector): boolean
IVecLen
+vecLen(IVector): double
VecLenEuclid
+vecLen(IVector): double
Point
-serialVersionUID: long
+Point(IVector)
IPlacedVector
+getBegin(): IVector
+getEnd(): IVector
Vector
-serialVersionUID: long
-coordinates: double[]
+Vector(double[])
+getCoordinates(): double[]
-varAdd(IVector, boolean): void
+add(IVector): void
+subtract(IVector): void
+scale(double): void
+isCompatible(IVector): boolean

Implementacja

Listing: IVector.java

```
1  package geometry7;
2  import java.io.Serializable;
3  public interface IVector extends Serializable {
4
5    public double[] getCoordinates();
6    public void add(IVector vec)throws IllegalArgumentException;
7    public void subtract(IVector vec)throws IllegalArgumentException;
8    public void scale(double factor);
9    public boolean isCompatible(IVector vec);
10
11 }
```

Implementacja

Listing: Vector.java

```
1  package geometry7;
2  import java.util.Arrays;
3  public class Vector implements IVector {
4    private static final long serialVersionUID = 4053383663313736838L;
5    private double[] coordinates;
6
7    public Vector(double[] coords) {coordinates = Arrays.copyOf(coords, coords.length); }
8    @Override
9    public double[] getCoordinates() { return Arrays.copyOf(coordinates, coordinates.length); }
10
11   private void varAdd(IVector vec, boolean add) throws IllegalArgumentException{
12     if(! isCompatible(vec))throw new IllegalArgumentException("Incompatible vectors");
13     double[] tmpCords = vec.getCoordinates();
14     for(int i=0;i<coordinates.length;i++)
15       coordinates[i]+=add? tmpCords[i]:(-tmpCords[i]);
16   }
17   @Override
18   public void add(IVector vec)throws IllegalArgumentException {varAdd(vec, true);}
19   @Override
20   public void subtract(IVector vec)throws IllegalArgumentException {varAdd(vec, false);}
```

Implementacja

Listing: Vector.java cd

```
21    @Override
22    public void scale(double factor) {
23      for(int i=0;i<coordinates.length;i++)
24        coordinates[i]*=factor;
25    }
26    @Override
27    public boolean isCompatible(IVector vec) {
28      if(coordinates.length != vec.getCoordinates().length)return false;
29      return true;
30    }
31  }
```

Implementacja

Listing: IPlacedVector.java

```
1 package geometry7;
2
3 public interface IPlacedVector extends IVector {
4   public IVector getBegin();
5   public IVector getEnd();
6 }
```

Implementacja

Listing: PlacedVector.java

```
1  package geometry7;
2
3  import java.util.Arrays;
4
5  import org.apache.commons.lang3.SerializationUtils;
6
7  public class PlacedVector implements IPlacedVector {
8
9    private static final long serialVersionUID = -8338265136413138187L;
10   protected IVector begin;
11   protected IVector trans;
12
13   public PlacedVector(IVector begin, IVector trans) {
14     if(!begin.isCompatible(trans)) throw new IllegalArgumentException("Incompatible Vectors");
15     this.begin = SerializationUtils.clone(begin);
16     this.trans = SerializationUtils.clone(trans);
17   }
18   @Override
19   public double[] getCoordinates() {  return Arrays.copyOf(trans.getCoordinates(),
       trans.getCoordinates().length);}
```

Implementacja

Listing: PlacedVector.java cd

```
20    @Override
21    public void add(IVector vec) throws IllegalArgumentException { trans.add(vec);  }
22    @Override
23    public void subtract(IVector vec) throws IllegalArgumentException {trans.subtract(vec); }
24    @Override
25    public void scale(double factor) { trans.scale(factor); }
26    @Override
27    public boolean isCompatible(IVector vec) {  return trans.isCompatible(vec); }
28    @Override
29    public IVector getBegin() { return SerializationUtils.clone(begin); }
30    @Override
31    public IVector getEnd() {
32      IVector tmp = SerializationUtils.clone(begin);
33      tmp.add(trans);
34      return tmp;
35    }
36
37  }
```

Implementacja

Listing: Point.java

```
1  package geometry7;
2
3  import org.apache.commons.lang3.SerializationUtils;
4
5  public class Point extends PlacedVector {
6    private static final long serialVersionUID = -8516592942676111319L;
7    public Point(IVector trans) {
8      super(SerializationUtils.clone(trans), SerializationUtils.clone(trans));
9      this.begin.subtract(trans);
10   }
11 }
```

Implementacja

Listing: IVecLen.java

```
1 package geometry7;
2
3 public interface IVecLen {
4   public double vecLen(IVector vec);
5
6 }
```

Implementacja

Listing: VecLenEuclid.java

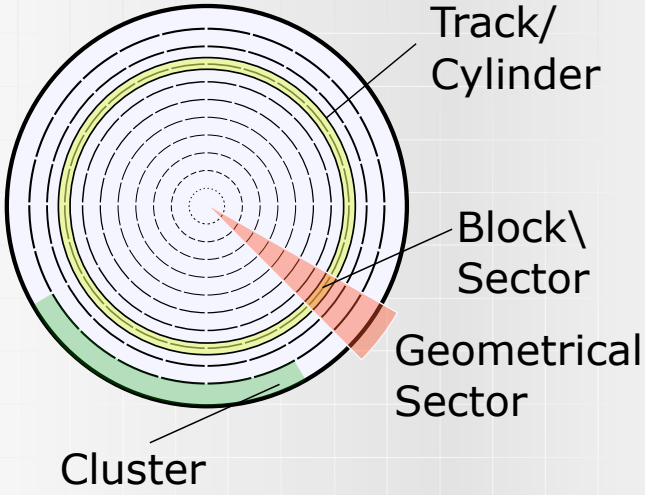
```
1  package geometry7;
2
3  public class VecLenEuclid implements IVecLen {
4
5    @Override
6    public double vecLen(IVector vec) {
7      double[] coords = vec.getCoordinates();
8      double sSum=0;
9      for(int i=0;i<coords.length;i++)
10       sSum+=coords[i]*coords[i];
11     return Math.sqrt(sSum);
12   }
13
14 }
```

Politechnika Wrocławska

Section 3

Modelujemy dysk twardy

Adresowanie danych na dysku



Kontroler RAID

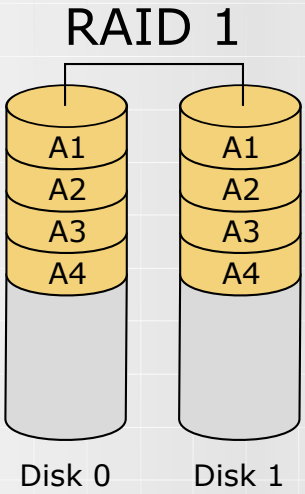
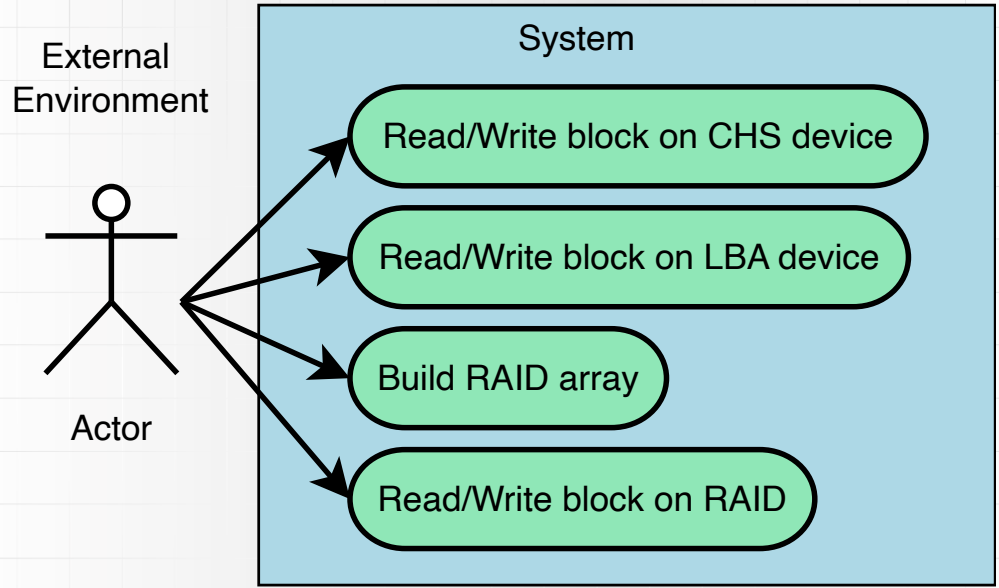
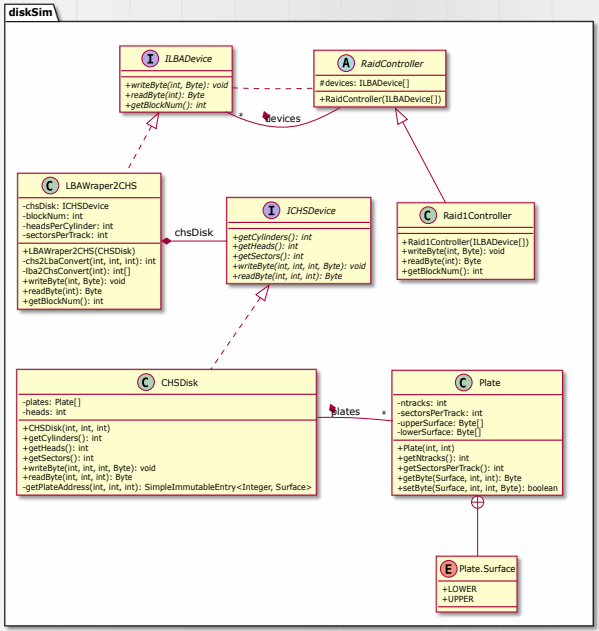


Diagram przypadków użycia




diskSim
ILBADevice
+writeByte(int, Byte): void
+readByte(int): Byte
+getBlockNum(): int
RaidController
#devices: ILBADevice[]
+RaidController(ILBADevice[])
devices
LBAWraper2CHS
-chsDisk: ICHSDevice
-blockNum: int
-headsPerCylinder: int
-sectorsPerTrack: int
+LBAWraper2CHS(CHSDisk)
-chs2LbaConvert(int, int, int): int
-lba2ChsConvert(int): int[]
+writeByte(int, Byte): void
+readByte(int): Byte
+getBlockNum(): int
chsDisk
ICHSDevice
+getCylinders(): int
+getHeads(): int
+getSectors(): int
+writeByte(int, int, int, Byte): void
+readByte(int, int, int): Byte
Raid1Controller
+Raid1Controller(ILBADevice[])
+writeByte(int, Byte): void
+readByte(int): Byte
+getBlockNum(): int
CHSDisk
-plates: Plate[]
-heads: int
+CHSDisk(int, int, int)
+getCylinders(): int
+getHeads(): int
+getSectors(): int
+writeByte(int, int, int, Byte): void
+readByte(int, int, int): Byte
-getPlateAddress(int, int, int): SimpleImmutableEntry<Integer, Surface>
plates
Plate
-ntracks: int
-sectorsPerTrack: int
-upperSurface: Byte[]
-lowerSurface: Byte[]
+Plate(int, int)
+getNtracks(): int
+getSectorsPerTrack(): int
+getByte(Surface, int, int): Byte
+setByte(Surface, int, int, Byte): boolean
Plate.Surface
+LOWER
+UPPER

Implementacja

Listing: Plate.java

```
1  package diskSim;
2
3  public class Plate {
4    private int ntracks;
5    private int sectorsPerTrack;
6    private Byte[][] upperSurface;
7    private Byte[][] lowerSurface;
8    public enum Surface{LOWER, UPPER}
9
10   public Plate(int ntracks, int sectorsPerTrack) {
11     this.ntracks = ntracks;
12     this.sectorsPerTrack = sectorsPerTrack;
13     upperSurface = new Byte[ntracks][sectorsPerTrack];
14     lowerSurface = new Byte[ntracks][sectorsPerTrack];
15     for(int i=0;i<ntracks;i++)
16       for(int j=0;j<sectorsPerTrack;j++) {
17         lowerSurface[i][j]=new Byte((byte) 0);
18         upperSurface[i][j]=new Byte((byte) 0);
19       }
20   }
21   public int getNtracks() {return ntracks;}
22   public int getSectorsPerTrack() {return sectorsPerTrack;}
```

Implementacja

Listing: Plate.java cd

```
23  public Byte getByte(Plate.Surface surface, int track, int sector) {
24    switch (surface) {
25    case LOWER:
26        return lowerSurface[track][sector].byteValue();
27    case UPPER:
28        return upperSurface[track][sector].byteValue();
29    default:
30      return null;
31    }
32  }
33  public boolean setByte(Plate.Surface surface, int track, int sector, Byte sbyte) {
34    switch (surface) {
35    case LOWER:
36        lowerSurface[track][sector] = new Byte(sbyte.byteValue());
37        return true;
38    case UPPER:
39        upperSurface[track][sector] = new Byte(sbyte.byteValue());
40        return true;
41    default:
42      return false;
43    }
44  }
```

Politechnika Wrocławska

Implementacja

Listing: ICHSDevice.java

```
1  package diskSim;
2
3  public interface ICHSDevice {
4
5    public int getCylinders();
6    public int getHeads();
7    public int getSectors();
8    public void writeByte(int C, int H, int S,Byte sbyte);
9    public Byte readByte(int C, int H, int S);
10
11 }
```

Implementacja

Listing: ILBADevice.java

```
1  package diskSim;
2
3  public interface ILBADevice {
4
5    public void writeByte(int lba,Byte sbyte);
6
7    public Byte readByte(int lba);
8
9    public int getBlockNum();
10 }
```

Implementacja

Listing: LBAWraper2CHS.java

```
1  package diskSim;
2
3  public class LBAWraper2CHS implements ILBADevice {
4
5    private ICHSDevice chsDisk;
6    private int blockNum;
7    private int headsPerCylinder;
8    private int sectorsPerTrack;
9
10   public LBAWraper2CHS(CHSDisk chsDisk) {
11     this.chsDisk = chsDisk;
12     headsPerCylinder = chsDisk.getHeads();
13     sectorsPerTrack = chsDisk.getSectors();
14     blockNum = chs2LbaConvert(chsDisk.getCylinders()-1, headsPerCylinder-1, sectorsPerTrack-1) +
        1;
15   }
16   private int chs2LbaConvert(int C, int H, int S) {
17     return (C*headsPerCylinder + H)*sectorsPerTrack + S;
18   }
```

Implementacja

Listing: LBAWraper2CHS.java cd

```
19  private int[] lba2ChsConvert(int lba) {
20    int[] chs = new int[3];
21    chs[0] = lba/(headsPerCylinder*sectorsPerTrack);
22    chs[1] = (lba/sectorsPerTrack) % headsPerCylinder;
23    chs[2] = lba % sectorsPerTrack;
24    return chs;
25  }
26
27  @Override
28  public void writeByte(int lba, Byte sbyte) {
29    int[] chs = lba2ChsConvert(lba);
30    chsDisk.writeByte(chs[0], chs[1], chs[2], sbyte);
31  }
32
33  @Override
34  public Byte readByte(int lba) {
35    int[] chs = lba2ChsConvert(lba);
36    return chsDisk.readByte(chs[0], chs[1], chs[2]);
37  }
```

Implementacja

Listing: LBAWraper2CHS.java cd

```
39    @Override
40    public int getBlockNum() {
41      return blockNum;
42    }
43
44
45  }
```

Politechnika Wrocławska

Implementacja

Listing: RaidController.java

```
1 package diskSim;
2
3 public abstract class RaidController implements ILBADevice {
4   protected ILBADevice[] devices;
5   public RaidController(ILBADevice[] devices) {this.devices=devices;}
6 }
```

Implementacja

Listing: Raid1Controller.java

```
1  package diskSim;
2
3  public class Raid1Controller extends RaidController {
4
5    public Raid1Controller(ILBADevice[] devices) {super(devices);}
6
7    @Override
8    public void writeByte(int lba, Byte sbyte) {
9      for(int i=0;i<devices.length;i++)devices[i].writeByte(lba, sbyte);
10   }
11   @Override
12   public Byte readByte(int lba) {   return devices[0].readByte(lba);  }
13
14   @Override
15   public int getBlockNum() {  return devices[0].getBlockNum(); }
16
17 }
```

Politechnika Wrocławska

Section 4

Małe podsumowanie

Politechnika Wrocławska

Czego już się nauczliśmy

- SOLID:
  - S Single Responsibility Principle
  - O Open–close principle
  - L Liskov substitution principle
  - I Interface segregation principle
  - D Dependency inversion principle
- Keep it simple, stupid. (KISS)
- Don’t Repeat Yourself (DRY)
- G.R.A.S.P (General Responsibility Assignment Software Principles)
  - Creator
  - Cohesion
  - Information Expert
  - Low coupling
  - Polymorphism
  - Protected variations
  - Pure fabrication
  - Indirection

Projektowanie Obiektowe 3

dr inż. Paweł Trajdos

Politechnika Wrocławska, Katedra Systemów i Sieci Komputerowych
Wyb. Wyspianskiego 27, 50-370 Wroclaw

5 lutego 2023