Aufgabenblatt: OOP II

(1.) Gegeben ist nebenstehende die Klasse CPoint!

(a.) Schreiben Sie eine Subklasse CPoint3 mit nebenstehenden Eigenschaften und Methoden und Testen Sie diese bei gegebenen Hauptprogramm TestCPoint3D!

(b.) Erstellen Sie bei den gegebenen Klassen CPoint und CPoint3D folgendes Hauptprogramm:

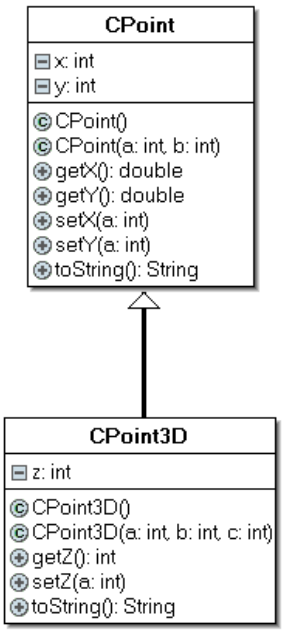
- Erstellen Sie eine neue Klasse CPoint3 mit den Werten 1,2,3

- Geben Sie die alle Werte über die Methode toString aus!

- Geben Sie x, y und z aus!

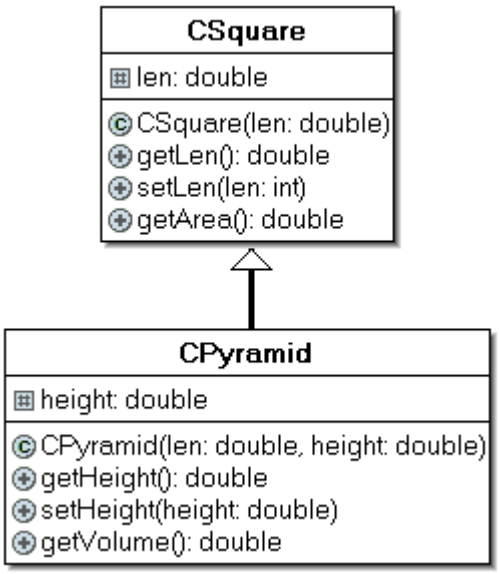
- Setzen Sie x,y und z mit Set… auf 4,5,6!

- Geben Sie mit toString alle Werte aus!



(2.) Geometrie:
Erstellen Sie eine Klasse CPyramid, die sich von der Klasse Square ableitet:

```
public class Quadrat {
  protected double len;
  Quadrat( double len ) {
    this.len = len;
  }
  public double getLen() {
    return len;
  }
  public void setLen(double len) {
    this.len = len;
  }
  public double getArea() {
    return len * len;
  }
}
```



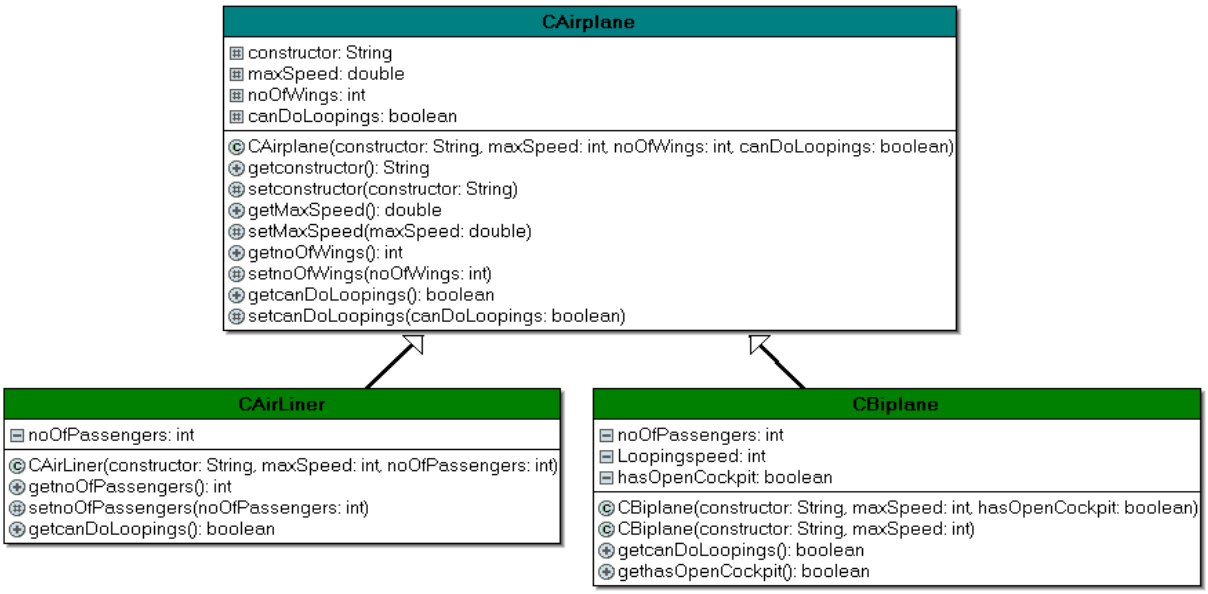
(3.) Im Rahmen einer Softwareentwicklung für die einheitliche Verwaltung von Flugzeugen wurde die Klasse CAirplane entwickelt. Die Klasse CAirplane ist wie folgt implementiert:

```
public class CAirplane {
  protected String constructor;
  protected double maxSpeed;
  protected int noOfWings;
  protected boolean canDoLoopings;
  public CAirplane(String constructor, int maxSpeed, int
noOfWings,boolean canDoLoopings) {
    this.constructor = constructor;
    this.maxSpeed = maxSpeed;
    this.noOfWings = noOfWings;
    this.canDoLoopings = canDoLoopings;
  }

  public String getconstructor(){
    return constructor;
  }
  protected void setconstructor(String constructor){
    this.constructor = constructor;
  }
  public double getMaxSpeed(){
    return maxSpeed;
  }
  protected void setMaxSpeed(double maxSpeed){
    this.maxSpeed = maxSpeed;
  }
  public int getnoOfWings(){
    return noOfWings;
  }
  protected void setnoOfWings(int noOfWings){
    this.noOfWings = noOfWings;
  }
  public boolean getcanDoLoopings(){
    return canDoLoopings;
  }
  protected void setcanDoLoopings(boolean canDoLoopings){
    this.canDoLoopings = canDoLoopings;
  }
}
```

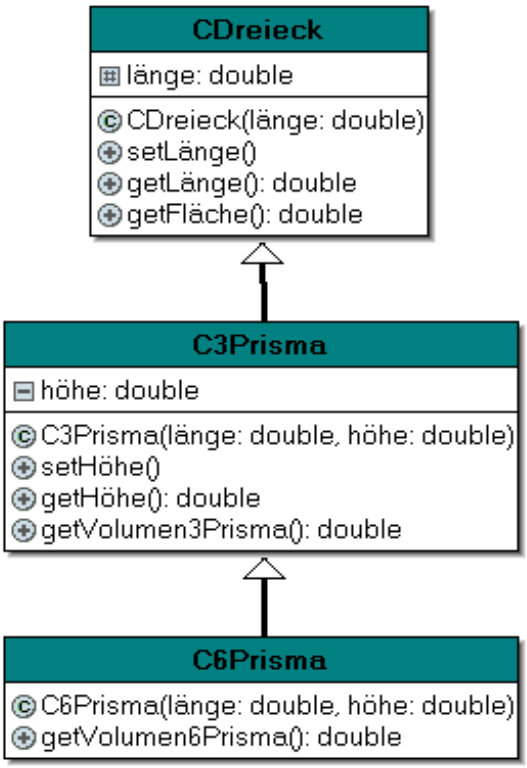
(a.) Schreiben Sie ausgehend von dieser Klasse eine Klasse CAirLiner(Verkehrsflugzeug). Diese sollen folgende Spezifikation erfüllen:
• Ein Verkehrsflugzeug ist ein Flugzeug, das genau ein Flügelpaar sowie eine zusätzliche Variable für die Anzahl der Passagiere(noOfPassengers) hat.
• Ein Verkehrsflugzeug fliegt keine Loopings. Stellen Sie deshalb sicher, dass die Methode getcanDoLoopings immer false zurückgibt!
•Schreiben Sie einen Konstruktor, mit dem ein Verkehrsflugzeug-Objekt erzeugt werden kann. Dazu müssen der Hersteller (String), die maximale Geschwindigkeit (int), und die Anzahl Passagiere (int) angegeben werden.
• Die Klasse Verkehrsflugzeug hat die Methoden getnoOfPassengers und setnoOfPassengers zum Abfragen und Setzen der Anzahl der Passagiere

(b.) Schreiben Sie ausgehend von dieser Klasse eine Klasse CBiplane(Doppeldecker). Diese sollen folgende Spezifikation erfüllen:
• Ein Doppeldecker ist ein Flugzeug, das genau zwei Flügelpaare hat.
• Weiter ist ein Doppeldecker akrobatiktauglich, d.h. man kann damit Loopings fliegen.
Für einen Looping muss der Doppeldecker eine Mindestgeschwindigkeit von 320 km/h erreichen. Definieren Sie dafür eine Konstante Loopingspeed. Die Methode getcanDoLoopings soll true zurückgeben, falls die zu lässige max. Geschwindigkeit(maxSpeed) grösser Loopingspeed ist.
• Die Klasse Doppeldecker hat eine Variable hasOpenCockpit vom Typ boolean. Sie gibt an, ob der Doppeldecker ein offenes oder geschlossenes Cockpit hat. Nachdem hasOpenCockpitgesetzt worden ist, darf sie nicht mehr verändert werden.
• Schreiben Sie eine Methode gethasOpenCockpit, die den Wert von hasOpenCockpitzurückgibt.
• Schreiben Sie zwei Konstruktoren, mit denen ein Doppeldecker-Objekt initialisiert werden kann.
o Der 1. Konstruktor hat folgende Parameter:Hersteller (String), maximale Geschwindigkeit (int) und einen boolean offenesCockpit, der angibt, ob der Doppeldecker ein offenes oder geschlossenes Cockpit hat
o Der 2. Konstruktor hat folgende Parameter:Hersteller (String), maximale Geschwindigkeit (int). Der Defaultwert für offenesCockpit bei einem Doppeldecker ist true
• Die Klasse Doppeldecker soll nicht erweiterbar sein



(4.) Erstellen Sie zwei Klassen CQuickSort und CBubbleSort und ermitteln Sie das Laufzeitverhalten für verschiedene Arraygrößen beim Sortieren. Füllen Sie dazu das/die Arrays mit zufälligen Werten und ermitteln Sie die Zeit für die Sortierung!

(5.) (a.) Erstellen Sie eine neue Klasse C6Prisma, die sich von der Klasse CTriangle über eine weitere Klasse ableitet und das Volumen eines Prismas bestimmt, dessen Grundseite aus 6 gleichseitigen Dreiecken besteht!
(b.) Erstellen Sie anlog eine Ableitung von einer Klasse CQuadrat über eine weitere Klasse nach der Klasse CQuader!



(6.) Erstellen Sie die Klassen von untenstehenden Klassendiagramm mit einer Klasse TestFortbewegungsmittel!