Probeklausur OOP1

Aufgabe 1

Richtige Antwort ist: "**eine is-a Beziehung**"

Aufgabe 2

a.

Eine abstrakte Methode ist eine Methode, die in einer Klasse deklariert, aber nicht implementiert wird. Das bedeutet, dass sie keinen Körper hat und keine konkrete Funktionalität besitzt. Abstrakte Methoden werden in abstrakten Klassen oder Interfaces definiert und dienen als Platzhalter für spezifische Implementierungen in abgeleiteten Klassen.

b.

Eine abstrakte Klasse ist eine Klasse, die mindestens eine abstrakte Methode enthält. Abstrakte Klassen können sowohl abstrakte als auch konkrete Methoden haben. Eine abstrakte Klasse kann nicht direkt instanziiert werden, sondern dient als Basis für abgeleitete (konkrete) Klassen, die ihre abstrakten Methoden implementieren.

c.

Eine abstrakte Methode wird in einer abstrakten Klasse (oder einem Interface) deklariert, indem der Methodenkopf (Signatur) angegeben wird, aber der Methodenkörper (Implementierung) fehlt. Das sieht etwa so aus:

```
public abstract void methode();
```

d.

Der Hauptunterschied zwischen einer abstrakten Klasse und einer normalen Klasse besteht darin, dass eine abstrakte Klasse abstrakte Methoden enthalten kann, während eine normale Klasse nur konkrete Methoden haben kann. Eine abstrakte Klasse kann nicht direkt instanziiert werden, während eine normale Klasse instanziiert werden kann.

e

Der Unterschied zwischen einer abstrakten Klasse und einem Interface liegt in erster Linie in ihrer Verwendung und ihren Eigenschaften:

- Eine abstrakte Klasse kann sowohl abstrakte als auch konkrete Methoden haben. Sie kann Variablen und Konstruktoren enthalten. Eine Klasse kann nur von einer einzigen abstrakten Klasse erben.
- Ein Interface kann abstrakte, static und default Methoden enthalten, sowie static, final Variablen avber keine Konstruktoren und allgemeine Variablen. Eine Klasse kann mehrere Interfaces implementieren.
- Eine Klasse kann von einer abstrakten Klasse erben und gleichzeitig Interfaces implementieren.
- Interfaces bieten eine Möglichkeit, eine Klasse mit mehreren von unterschiedlichen "Interfaces" erben zu lassen, während abstrakte Klassen eher dazu dienen, gemeinsame Funktionalitäten zu teilen und abgeleitete Klassen zu strukturieren.
- Interfaces können von anderen Interfaces erben.

Aufgabe 3

Datei Druckerverwaltung:

```
public class Druckerverwaltung {
 public static void main(String[] args) {
```

```
  Drucker d1 = new Drucker("Canon", "abc100", 199); // <- Dieser Konstruktor
existiert nicht
  Laserdrucker d2 = new Laserdrucker("HP", "L 1001", 119.95, "xyz-Toner");
  Drucker d3 = new Laserdrucker("Brother", "B 2002", 99, "abc-Toner");

  System.out.println(d1.getTonerTyp()); // <- die Klasse Drucker besitzt diese
Methode nicht
  System.out.println(d2.getTonerTyp());
  System.out.println(d3.getTonerTyp()); // <- die Klasse Drucker besitzt diese
Methode nicht

  d2.versendeFax(+4912345);
 }
}
```

Datei Drucker.java:

```
public class Drucker {
 public String marke;
 private String bezeichnung;
 double preis;

 Drucker(){
  marke = "unbekannt";
  bezeichnung = "unbekannt";
  preis = 999;
 }

 void print() {
  System.out.println(marke + "//" + bezeichnung + "//" + preis);
 }
}
```

Datei Laserdrucker.java:

```
public class Laserdrucker extends Drucker extends KannFaxen{ // <- das zweite extends
müsste implements sein
 String tonerTyp;

 Laserdrucker(String marke, String bezeichnung, int preis, String tonerTyp){
  super(marke, bezeichnung, preis); // <- Dieser Konstruktor excistiert nicht
  this.marke = marke;
  this.bezeichnung = bezeichnung; // <- Die Variable ist private und es kann nicht
daruaf zugegriffen werden
  this.preis = preis;
  this.tonerTyp = tonerTyp;
 }

 public String getTonerTyp() {return tonerTyp; }

 public void versendeFax(int nummer) { System.out.println("Noch nicht
implementiert");}

 public void empfangeFax(int nummer) { System.out.println("Noch nicht
implementiert");}
}
```

Datei KannFaxen.java

```
public interface KannFaxen {
 void versendeFax(int nummer);
 void empfangeFax(int nummer);
}
```

Aufgabe 4