

Klausur – Objektorientierte Programmierung in Java

19. Nov. 2025 , Arbeitszeit: 90 Minuten

Aufgabe: Implementierung der Klasse Point

Im Unterricht haben Sie bereits verschiedene Klassen wie `Bruch` und `GiroKonto` implementiert. In dieser Klausur sollen Sie eine weitere Klasse entwickeln: `Point`.

Ein `Point` beschreibt einen Punkt in einem zweidimensionalen Koordinatensystem. Implementieren Sie die Klasse vollständig gemäß der folgenden Spezifikation:

1. Attribute (private)

- `int x` – x-Koordinate
- `int y` – y-Koordinate

2. Konstruktoren

- Ein Konstruktor, der beide Koordinaten als Parameter übernimmt.
- Ein Standardkonstruktor, der den Punkt auf (0, 0) setzt.

3. Getter- und Setter-Methoden

- `getX()`, `getY()`
- `setX(int x)`, `setY(int y)`

4. Weitere Methoden

- `public void move(int dx, int dy)` Verschiebt den Punkt um die angegebenen Werte.
- `public double distance(Point p)` Berechnet die Distanz zu einem anderen Punkt `p`:

$$\sqrt{(x - p.x)^2 + (y - p.y)^2}$$

- `public void mirrorX()` Spiegelt den Punkt an der **x-Achse**. (y-Wert wird negiert.)
- `public void mirrorY()` Spiegelt den Punkt an der **y-Achse**. (x-Wert wird negiert.)
- `public String toString()` Gibt den Punkt im Format "(x, y)" zurück.



5. Testprogramm

Schreiben Sie eine Klasse `PointTest` mit `main`-Methode, die:

- zwei Punkte erzeugt,
- beide Punkte ausgibt,
- einen Punkt verschiebt,
- die Distanz zwischen beiden Punkten berechnet und ausgibt,
- die Spiegelungen mit `mirrorX()` und `mirrorY()` demonstriert.

6. Dokumentation und Codequalität

Achten Sie bei der gesamten Implementierung auf:

- sinnvolle Bezeichner (Variablen-, Methoden- und Klassennamen),
- eine klare Klassenstruktur,
- JavaDoc-Kommentare für:
 - die Klasse `Point`,
 - alle Konstruktoren,
 - alle Methoden,
- aussagekräftige Inline-Kommentare an sinnvollen Stellen,
- korrekte Sichtbarkeiten (Kapselung: Attribute `private`),
- übersichtliche Formatierung (Einrückungen, Zeilenumbrüche).

7. Bonus-/Zusatzaufgabe (optional, Empfehlung: 5 Punkte)

Erweitern Sie die Klasse `Point` um die Methode:

- `public boolean equals(Point p)` Prüft, ob zwei Punkte gleiche Koordinaten besitzen.

Geben Sie `true` zurück, wenn beide Punkte identisch sind, sonst `false`



Bewertungsschema (100 BE)

| | | |
|----------------------------------|---|-------|
| 1. Attribute | private, richtige Datentypen | 8 BE |
| 2. Konstruktoren | Standardkonstruktor + 2- Parameter-Konstruktor | 15 BE |
| 3. Getter & Setter | alle Getter und Setter korrekt umgesetzt | 10 BE |
| 4. move() | korrekte Verschiebung des Punktes | 5 BE |
| 5. mirrorX() | Spiegelung an der x-Achse (y wird negiert) | 4 BE |
| 6. mirrorY() | Spiegelung an der y-Achse (x wird negiert) | 4 BE |
| 7. distance() | korrekte Formel, Datentyp double, richtige Berechnung | 18 BE |
| 8. toString() | Ausgabe im Format „(x, y)“ | 5 BE |
| 9. Testprogramm | Objekte erzeugen, ausgeben, verschieben, Distanz & Spiegelungen demonstrieren | 18 BE |
| 10. Dokumentation & Codequalität | JavaDoc, Kommentare, Struktur, Einrückung, sinnvolle Namen | 13 BE |

Optional: Bonusaufgabe equals() (5 BE zusätzlich)

Auswertung

| Aufgabe | max. BE | erreicht |
|----------------------------------|------------|----------|
| 1. Attribute | 8 | |
| 2. Konstruktoren | 15 | |
| 3. Getter & Setter | 10 | |
| 4. move() | 5 | |
| 5. mirrorX() | 4 | |
| 6. mirrorY() | 4 | |
| 7. distance() | 18 | |
| 8. toString() | 5 | |
| 9. Testprogramm | 18 | |
| 10. Dokumentation & Codequalität | 13 | |
| Gesamt | 100 | |

Note: _____



Viel Erfolg!