

13. Aufgabe (3 Punkte, Methoden ausführen, Verhalten analysieren)

```
class Bsp{
    int methA(int par){
        int x = par;
        x = x + 1;
        x = x + 2;
        x = x + 3;
        return x;
    }
    int methB(int par){
        int x = par;
        int y = x + x + x;
        y = y + y;
        x = y - x;
        return x;
    }
    int methC(int par1, int par2){
        int x = par1 + par1;
        int y = par2 + par2;
        y = y - par1;
        x = x - par2;
        int ergebnis = x + y;
        return ergebnis;
    }
}
```

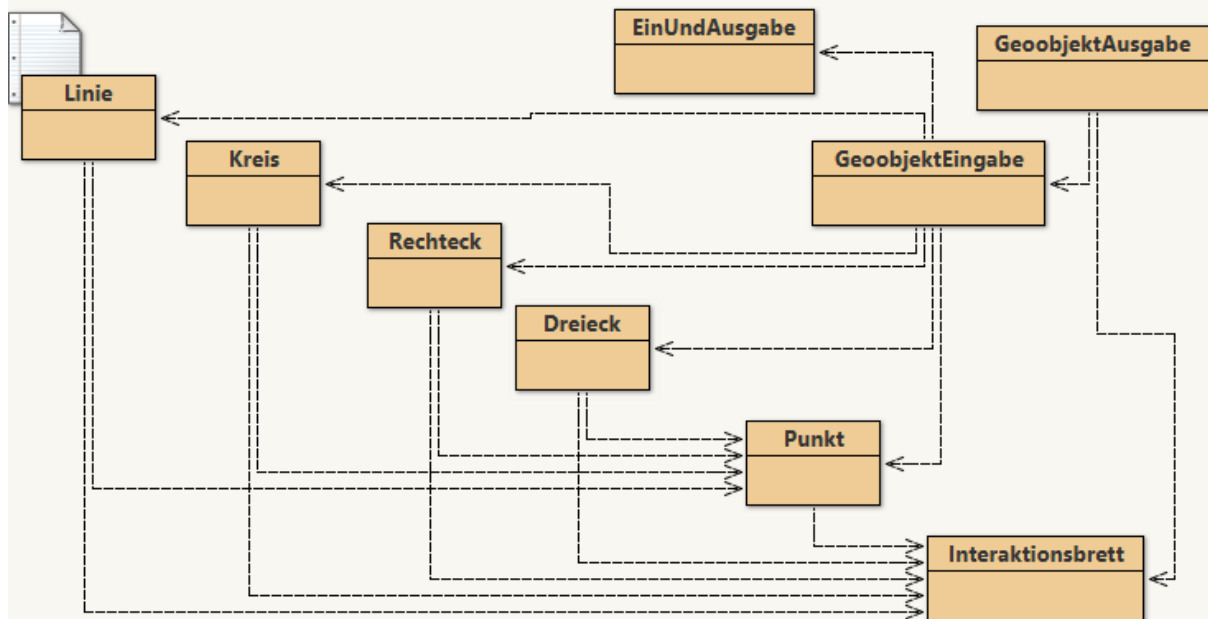
Challenge zur
Selbstein-
schätzung: Sie
haben 8
Minuten
Aufgabe a) und
c) auf Papier zu
lösen; klappt
das???

Gegeben sei die Klasse Bsp (s. auch Webseite).

- Überlegen und notieren Sie, welches Ergebnis zurückgegeben wird, wenn für ein Objekt b vom Typ Bsp die Aufrufe b.methA(6), b.methB(5) und b.methC(2,2) macht werden.
 - Führen Sie mit dem Code Pad die notwendigen Schritte aus, um ihre Annahmen aus a) überprüfen zu können und machen Sie dazu passende Bildschirmfotos.
- c) Die folgenden Befehle werden nacheinander im Code Pad ausgeführt, geben Sie rechts daneben an, ob es sich bei einer Zeile um eine Deklaration, eine Zuweisung, einen Ausdruck oder um syntaktischen Unsinn, also einen Fehler, handelt. Falls nach der Zeile eine Ausgabe erfolgt, geben Sie diese auch an.

Zeile	Dekla- ration?	Zuwei- sung?	Aus- druck?	Fehler?	Ausgabe
Bsp b;					
b = new Bsp();					
int a:a;					
int b;					
int c = b.methB(2);					
methA(c);					
Bsp d = new Bsp();					
int e = d.methC(c,c);					
e					
d.methA(c) = 42;					
d.methA(c);					
d.methA(c)					
c+c					
c+c;					
d+d					
c+d;					
d.methC(c,c) + c					

14. Aufgabe (9 Punkte, Methoden nutzen Methoden, Ein- und Ausgabe)



Gegeben seien Ihre Klassen Punkt, Linie, Kreis, Rechteck und Dreieck.

- a) Ergänzen Sie in jeder Ihrer Klassen eine Methode `void darstellen(Interaktionsbrett ib)` mit der sich das jeweilige Objekt auf dem übergebenen Interaktionsbrett zeichnet. Nutzen Sie dabei Ihre `get`-Methoden, um die Parameter für die Zeichenmethoden von `ib` zu erhalten.

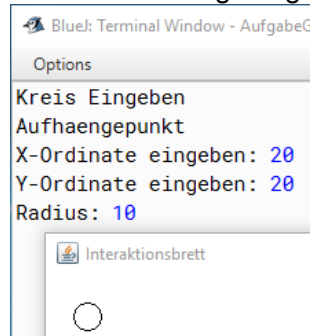
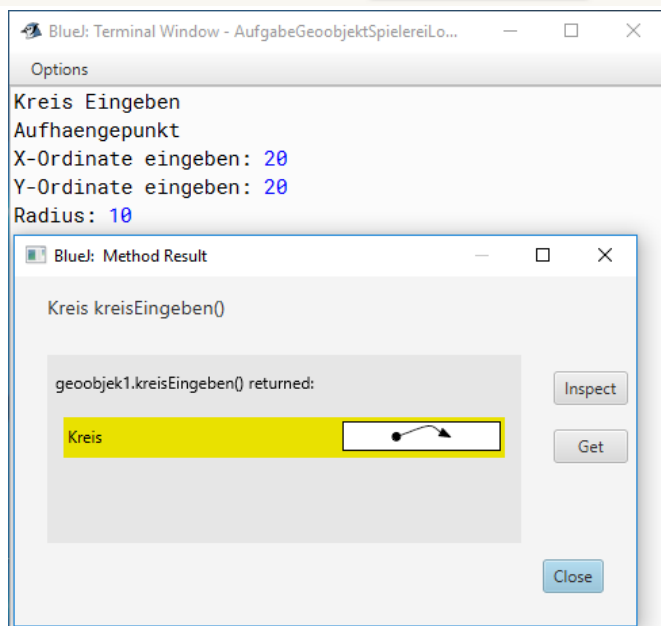
- b) Ergänzen Sie eine Klasse `GeobjektEingabe` mit einer Objektvariablen vom Typ `EinUndAusgabe`. Schreiben Sie in dieser Klasse für jede der fünf oben genannten Klassen jeweils eine Methode, mit der die

nutzende Person aufgefordert wird die jeweils notwendigen Daten einzugeben und die dann die jeweiligen Objekte als Ergebnis zurückliefert. Eine der geforderten Methoden von `GeobjektEingabe` sieht damit wie folgt aus: `Kreis kreisEingeben() { ...`

Wird diese Methode aufgerufen, sollte der Dialog wie in der Mitte rechts gezeigt aussehen (Eingaben sind blau, weiterhin ist das zurückgegebene Objekt sichtbar).

Überlegen Sie sich, wie man das Kopieren von Programmcodezeilen beim Erstellen der benötigten Punkte verhindern kann.

- c) Ergänzen Sie eine Klasse `GeobjektAusgabe` mit jeweils einer Objektvariablen vom Typ `Interaktionsbrett` und `GeobjektEingabe`. Schreiben Sie in dieser Klasse für jede der fünf oben genannten Klassen jeweils eine Methode, mit der der Benutzer aufgefordert wird ein Objekt einzugeben und



die dann die jeweiligen Objekte mit Hilfe der Methoden aus a) in dem Interaktionsbrett zeichnet. Eine der Methoden von GeoobjektAusgabe sollte wie folgt aussehen: void kreisAusgeben() { ...

Wird dann diese Methode für ein Objekt der Klasse GeoobjektAusgabe aufgerufen, erscheint zunächst der Dialog aus b) und dann die auf der vorherigen Seite unten rechts gezeigte Ausgabe.

15. Aufgabe (2 Punkte, Nutzung des Debuggers)

Gegeben sei die auf der rechten Seite auch von der Veranstaltungsseite erhältliche Klasse. Analysieren Sie die Methoden wasPassiert1 und wasPassiert2 mit dem Debugger von BlueJ. Markieren Sie dazu jeweils die erste Zeile der Methoden als Breakpoint. Erzeugen Sie dann ein Objekt der Klasse AnalysierMich und rufen Sie die Methoden einzeln auf. Fotografieren (screenshotten) Sie jeden Schritt des Debuggers während der Ausführung der Methode. Das untere Bild zeigt z. B. die Ausgangssituation unmittelbar nach dem Start von wasPassiert1, nachdem ein Objekt AnalysierMich(42,17) erzeugt wurde. Sie dürfen die Fenster natürlich anders anordnen.

Was vermuten Sie, welchen Effekt haben die beiden Methoden?

```
class AnalysierMich{
    int kraft;
    int iq;

    AnalysierMich(int kraft, int iq){
        this.kraft = kraft;
        this.iq = iq;
    }

    void wasPassiert1(){
        this.iq = this.kraft + this.iq;
        this.kraft = this.iq - this.kraft;
        this.iq = this.iq - this.kraft;
    }

    void wasPassiert2(){
        this.kraft = this.kraft + this.kraft;
        int tmp = this.kraft;
        this.kraft = this.kraft + this.kraft;
        this.kraft = this.kraft + tmp;
    }
}
```

The image shows a screenshot of a Java IDE. On the left, a code editor displays the class `AnalysierMich` with three methods: `AnalysierMich(int kraft, int iq)`, `wasPassiert1()`, and `wasPassiert2()`. A red arrow points to the first line of `wasPassiert1()`. On the right, the BlueJ Debugger window is open, showing the 'Call Sequence' pane with `AnalysierMich.wasPassiert1` selected. The 'Instance variables' pane shows `int kraft = 42` and `int iq = 17`.

16. Aufgabe (4 Punkte, Methoden rufen Methoden auf, Klausurähnlichkeit)

Gegeben seien folgende Klassen. Arbeiten Sie die Methode `gruppenerlebnis()` schrittweise ab und zeichnen sie ab der vierten Zeile nach jeder Zeile ein Objektspeicherbild, wie sie auch in der Vorlesung verwandt werden. Vereinfachend ist das Bild nach der dritten Zeile bereits angegeben (falls Sie es mit Powerpoint bearbeiten wollen, ist die Datei auf der Veranstaltungsseite).

```
class Studierend{
    String vorname = "Eva";
    String nachname = "Mustermann";
    int matrikelnummer = 232323;

    Studierend(String vorname, String nachname
        , int matrikelnummer) {
        this.vorname = vorname;
        this.nachname = nachname;
        this.matrikelnummer = matrikelnummer;
    }

    String getVorname() {
        return vorname;
    }

    void setVorname(String vorname) {
        this.vorname = vorname;
    }

    String getNachname() {
        return nachname;
    }

    void setNachname(String nachname) {
        this.nachname = nachname;
    }

    int getMatrikelnummer() {
        return matrikelnummer;
    }

    void setMatrikelnummer(int mat) {
        this.matrikelnummer = mat;
    }
}
```

```
class Arbeitsgruppe{
    Studierend mitglied1;
    Studierend mitglied2;

    Arbeitsgruppe(Studierend p1, Studierend p2){
        this.mitglied1 = p1;
        this.mitglied2 = p2;
    }

    Studierend getMitglied1(){
        return this.mitglied1;
    }

    Studierend getMitglied2(){
        return this.mitglied2;
    }

    void setStudierend1(Studierend s){
        this.mitglied1 = s;
    }

    void setStudierend2(Studierend s){
        this.mitglied2 = s;
    }
}

class ArbeitsgruppenAnalyse{
    Studierend gruppenerlebnis(){
        Studierend s1 = new Studierend("Bo","Li",42);
        Studierend s2 = new Studierend("Jo","Mi",43);
        Studierend s3 = new Studierend("Mo","Ma",44);
        // ab hier nach jedem Befehl zeichnen
        Arbeitsgruppe a1 = new Arbeitsgruppe(s1, s2);
        Arbeitsgruppe a2 = new Arbeitsgruppe(s2, s3);
        Studierend tmp = a1.getMitglied2();
        tmp.setVorname("Hein");
        a2.getMitglied1().setNachname("Mett");
        a1.getMitglied2()
            .setMatrikelnummer(s3.getMatrikelnummer());
        return a1.getMitglied2();
    }
}
```

Ausgangssituation nach drei Zeilen:

