Punkte ab	Note	Anzahl	Prozent
0	5,0	27	37 %
45	4,0	10	14 %
50	3,7	7	10 %
55	3,3	6	8 %
60	3,0	3	4 %
65	2,7	5	7 %
70	2,3	2	3 %
75	2,0	3	4 %
80	1,7	3	4 %
85	1,3	6	8 %
90	1,0	1	1 %

Aufgabe 1 (10 P)
Aus der Mathematik wissen wir, dass die unendliche Summe

$$\sum_{k=1}^{\infty} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots$$

nicht konvergiert, das heißt es wird jede beliebige Zahl als Summe erreicht oder überschritten, wenn man nur genügend Summanden addiert.

Implementieren Sie eine (statische) Methode, die für eine einzugebende Grenze s die Anzahl der Folgeglieder bestimmt, die addiert werden müssen, um s (erstmalig) zu erreichen oder zu übersteigen.

Beispiel:
Für $s = 2$ sollte die Methode die Anzahl $n = 4$ liefern, denn

$$\sum_{k=1}^{4} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = 2,08\overline{3} > 2 \quad \text{und} \quad \sum_{k=1}^{3} \frac{1}{k} = 1 + \frac{1}{2} + \frac{1}{3} < 2.$$

Überlegen Sie genau, welcher Schleifentyp hier angebracht ist!

Lösung 1
Do-While-Schleife oder while-Schleife sind angemessen, weil die Anzahl der benötigten Durschläufe nicht bekannt ist.

```
public static int HarmonischeReihe(double s) {
        int n = 0;
        double r = 0;
        while(r < s) {
                n++;
                r = r + 1.0/n;
        }
        return n;
}
```

Bewertung: Punktabzug für

- unpassender Methodenkopf (Eingabeparam int: ohne Punktabzug)
- unkorrekte Schleife
- Verwendung einer for-Schleife
- int-Division
- falsche Ergebnisse für Spezialfälle von s ($< 0, = 1$)
- fehlendes/falsches return

Aufgabe 2 (15 P)
Wir nehmen an, es sei ein double-Array gegeben, in dem die täglichen Inzidenz-Werte seit Beginn der Pandemie abgelegt seien. Schreiben Sie eine (statische) Methode, der ein double-Array übergeben wird, und die ein (neues) Array liefert, in dem für die letzen 30 Tage die jeweilige 7-Tage-Inzidenz, also der Durchschnitt der letzten 7 Werte abgelegt ist.

Lösung 2

```
public static double[] corona2(double[] a) {
        double[] erg = new double[30];
        int l = a.length;
        for (int i = 0; i < erg.length; i++) {
                // berechne 7-Tage-Inzidenz von Tag i
                double sum = 0;
                for (int j = 1; j <= 7; j++)
                        sum = sum + a[l+i-30-j];
                erg[i] = sum/7;
        }
        return erg;
}
```

Bewertung: Punktabzug für

- falschen Methodenheader
- fehlendes/falsches Ergebnis-Array (Definition und return)
- prinzipiell falsche Schleifenkonstruktion

Kein Punktabzug für maximal um 2 versetzte Grenzen, die auf anderes Verständnis von „einschließlich-ausschließlich“ zurückzuführen sind

Kontrollergebnis:
für a =$\{0, 1, 3, \ldots 36\}$ ergeben sich 7-Tage-Inzidenzen erg = $\{3, 4, \ldots 32\}$

Aufgabe 3 (10 P)
Stellen Sie eine Tracetabelle für das folgende Code-Stück auf. Gehen Sie dabei davon aus, dass die Variable n mit dem Wert der letzten Ziffer Ihrer Matrikelnummer belegt sei.

Notieren Sie explizit, mit welchen Parametern die Methode meth() aufgerufen wird.

```
1  public static void main(String[] args) {
2          int n;
3          // n = die letzte Ziffer Ihrer Matrikelnummer
4          int x = 2;
5          int y = 3;
6          if(n<5) {
7                  if(n>2)
8                          x = meth(x+1, y-1, n+4) + n;
9                  else
10                         y = meth(x+2, y+1, n+7) + n;
11         }
12         else {
13                 if(n>7)
14                         x = meth(x-1, y+1, n-1) + n;
15                 else
16                         y = meth(x-1, y-1, n+2) + n;
17         }
18 }
19
20 public static int meth(int a, int b, int n) {
21         for(int i = 0; i < n-5; i++) {
22                 a = a+b;
23                 b = b+2;
24         }
25         return a;
26 }
```

Lösung 3
Tabellen für die verschiedenen Matr-Nummern auf der folgenden Seite.
Methodenaufruf an erster Zeile in meth-Teil erkennbar

Bewertung: deutlicher Punktabzug für

- falsche Endzifffer für n
- falsche Tabellenstruktur (Spaltenüberschriften) / fehlende Spalten
- nicht erkennbare Übergabeparameter/falscher Methodenaufruf
- lokale Var n vs. Parameter n nicht nicht richtig behandelt
- falsches Schleifenende
- fehlende/falsche letzte Zeile der Tabelle

nur minimalen Punktabzug für Rechenfehler innerhalb der Schleife
keinen weiteren Punktabzug für Folgefehler

in main | in meth

n	x	y	a	b	n	i
0	2	3				
			4	4	7	0
			8	6		1
			14	8		2
	2	14				

in main | in meth

n	x	y	a	b	n	i
1	2	3				
			4	4	8	0
			8	6		1
			14	8		2
			22	10		3
	2	23				

in main | in meth

n	x	y	a	b	n	i
2	2	3				
			4	4	9	0
			8	6		1
			14	8		2
			22	10		3
			32	12		4
	2	34				

in main | in meth

n	x	y	a	b	n	i
3	2	3				
			3	2	7	0
			5	4		1
			9	6		2
	12	3				

in main | in meth

n	x	y	a	b	n	i
4	2	3				
			3	2	8	0
			5	4		1
			9	6		2
			15	8		3
	19	3				

in main | in meth

n	x	y	a	b	n	i
5	2	3				
			1	2	7	0
			3	4		1
			7	6		2
	2	12				

in main | in meth

n	x	y	a	b	n	i
6	2	3				
			1	2	8	0
			3	4		1
			7	6		2
			13	8		3
	2	19				

in main | in meth

n	x	y	a	b	n	i
7	2	3				
			1	2	9	0
			3	4		1
			7	6		2
			13	8		3
			21	10		4
	2	28				

in main | in meth

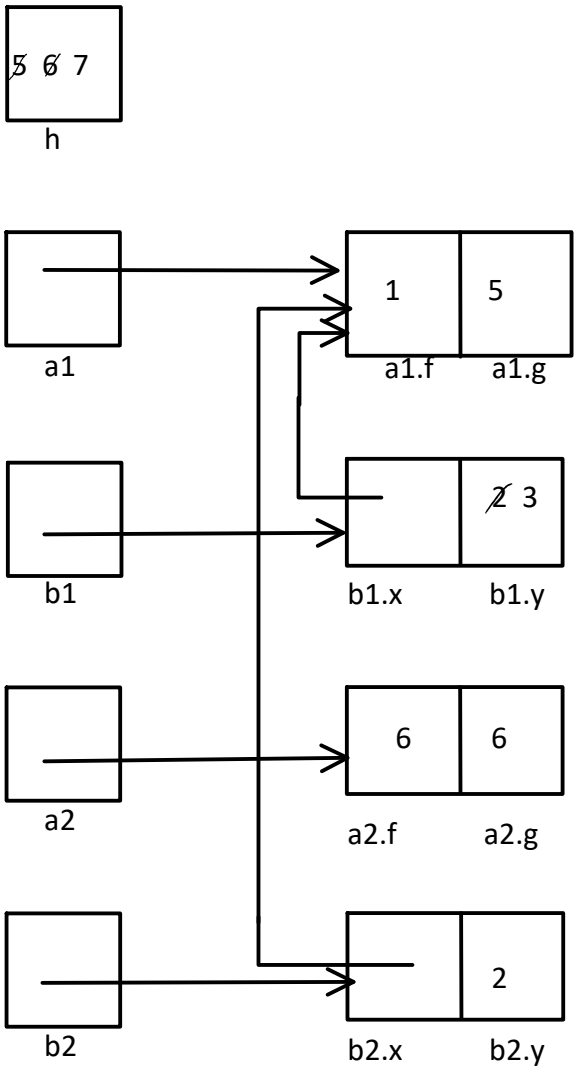
n	x	y	a	b	n	i
8	2	3				
			1	4	7	0
			5	6		1
			11	8		2
	19	3				

in main | in meth

n	x	y	a	b	n	i
9	2	3				
			1	4	8	0
			5	6		1
			11	8		2
			19	10		3
	28	3				

Aufgabe 4 (10 P)
Gegeben seien die folgenden Klassen. Zeichnen Sie die Speichersituation am Ende der main-Methode.

Lösung 4
Speicherbild:



Aufgabe 5 (7 + 3 + 5 P)

a) Implementieren Sie eine (statische) Methode `createWorte()`. Die Methode nimmt ein Array mit Zeichen und ein Array mit ganzzahligen Werten an. Die ganzzahligen Werte des zweiten Arrays dienen als Index für Zeichen des ersten Arrays: Die Methode soll ein Wort aus denjenigen Zeichen zusammensetzen, deren Nummer im zweiten Array angegeben sind. Das so entstandene Wort wird zurückgegeben.

Beispiel:
Der Aufruf der Methode mit den Arrays {'S','C','H','I','F','F','E'} und {0, 6, 3, 5, 6} liefert das Wort SEIFE.

b) Bei der Übergabe von „falschen" Arrays kann es zu Problemen kommen. Die Methode soll eine Exception auslösen, wenn ein solcher Fall eintritt.

c) Schreiben Sie eine (statische) Methode `createWorteTesten()` zum Testen Ihrer Methode. Implementieren Sie eine Anweisung mit einem beliebigen Character-Array und ungültigen Index-Werten. Fangen Sie die Exception durch einen Exception-Handler ab.

Lösung 5

```
public static String createWorte(char[] array, int[] indizes){
    String result ="";
    for(int i = 0; i<indizes.length; i++){
        int index = indizes[i];
        // Loesung zu b):
        if(index<0 || index>= array.length)
            throw new IndexOutOfBoundsException();
        // Ende Loesung zu b)
        result += array[index];
    }
    return result;
}

public static void createWorteTesten(){
   try {
        String wort = createWorte(new char[]{'S', 'C'},
                                  new int[]{0,-1, 10});
   }
   catch (ArrayIndexOutOfBoundsException e){
        System.out.println("falscher Index");
   }
}
```

Bewertung:

a)

b)

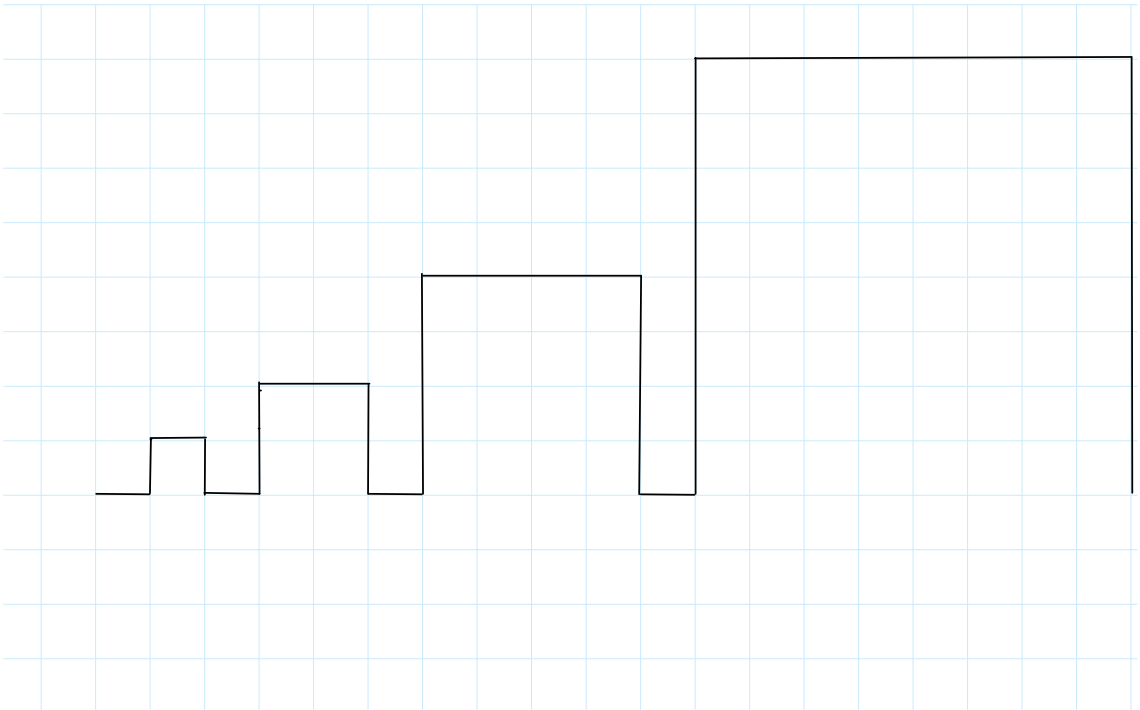
c)

Aufgabe 6 (10 P)
Gegeben sei das folgende Interface

```
public interface Turtle {

        void step(); // ein Schritt in Blickrichung

        void step(int n); // n Schritte in Blickrichtung

        void turnL(); // Vierteldrehung gegen den Uhrzeiger

        void turnR(); // Vierteldrehung mit dem Uhrzeiger
}
```

Nehmen Sie an, die Klasse `Roboter` implementiere dieses Interface.
Die Klasse verfüge über einen parameterlosen Konstruktor, der einen Roboter mit Startposition $(0, 0)$ und Blickrichtung „rechts“ erzeugt.

Implementieren Sie eine statische Methode `laufWeg(int n)`, in der ein Roboter-Objekt erzeugt wird. Der Roboter soll einen Weg laufen, der die unten gezeigte Gestalt hat. Der Parameter n gibt die Anzahl der Quadrate an. Im Bild ist der zurückgelegte Weg für den Parameterwert $n = 4$ gezeigt.



Lösung 6

```
public static void laufWeg(int n) {
        Roboter r2d2 = new Roboter();
        int a = 1; // Kantenlaenge des Quadrats
        for (int i = 1; i <= n; i++) {
                r2d2.step();
                r2d2.turnL();
                r2d2.step(a);
                r2d2.turnR();
                r2d2.step(a);
                r2d2.turnR();
                r2d2.step(a);
                r2d2.turnL();
                a = 2*a;
        }
}
```

Bewertung: Punktabzug für

- kein Roboter-Objekt innerhalb der Methode erzeugt
- falsche Anzahl Schleifendurchläufe
- falscher Schleifenbody
- Kantenlänge des Quadrats nicht verdoppelt
- print o.ä.
- Benutzung andere Methoden als der Methoden aus dem Interface

kein Punktabzug für andere Blickrichtung am Ende (als „rechts“)

Aufgabe 7 (3 + 10 + 7 P)

a) Definieren Sie ein Interface `Transportmittel` mit folgenden Eigenschaften: Transportmittel bieten Methoden:

- `istBuchbar()`: zur Bestimmung, ob ein Platz für eine Person, oder ein Paket buchbar/ frei ist oder nicht. Die Methode gibt einen Wahrheitswert zurück.
- `gesamtdauer()`: die Methode nimmt einen ganzzahligen Kilometerwert an und gibt einen Fliesskommazahlwert zurück, wie viel Zeit in Minuten vergeht bis die gesamte Strecke zurück gelegt ist.

b) Implementieren Sie eine abstrakte Klasse `Personentransport` mit folgenden Eigenschaften: Personentransporte sind Transportmittel und haben:

- eine Bezeichnung `bez` vom Typ `String`,
- eine ganzzahlige Gesamtsitzplatzanzahl `plaetze` vom Typ `int` und
- einen ganzzahligen Wert `belegt` vom Typ `int`, der angibt wieviele Plätze aktuell belegt sind.
- Die Bezeichnung und die Anzahl der Gesamtsitzplätze werden bei Erzeugung eines Objektes übergeben. Die Anzahl der belegten Sitzplätze wird bei Erzeugung auf 0 gesetzt.
- Implementieren Sie die Methode `istBuchbar()` sodass überprüft wird, ob noch Plätze frei sind.
- Personentransporte haben ausserdem eine `einsteigen()`- und eine `aussteigen()`-Methode, die die Anzahl der belegten Plätze entsprechend verändert.
- Die `toString()`-Methode soll Bezeichnung und Anzahl *freier* Plätze liefern.

c) Implementieren Sie eine Klasse `Bus` mit folgenden Eigenschaften: Busse sind Personentransporte und

- Busse haben einen Fliesskommazahlwert `dauer`, der die benötigte Zeit pro Kilometer in Minuten angibt.
- Implementieren Sie hier die Methode `gesamtdauer()` des Interface.
- Sehen Sie ausserdem eine `get`- und eine `set`- Methode für die Dauer vor.
- Bei Erzeugung des Objektes werden die Dauer und die Anzahl der Gesamtsitzplätze übergeben. Die Bezeichnung setzt sich aus dem Stichwort *Bus* und einer anschliessenden automatisch bestimmten Zahl zusammen. So bekommt der erste Bus die Bezeichnung *Bus1*, der zweite *Bus2*, usw.

Definieren Sie die Klassen mit geeigneten Klassen- und Instanzattributen, Konstruktor und den geforderten Methoden. Achten Sie auf Zugriffsrechte!

Lösung 7

```java
public interface Transportmittel {
    public boolean istBuchbar();
    public double gesamtdauer(int km);
}

public abstract class Personentransport implements Transportmittel {
    private String bez;
    private int plaetze;
    private int belegt;

    public Personentransport(String bez, int plaetze){
        this.bez = bez;
        this.plaetze = plaetze;
        this.belegt = 0;
    }
    public boolean istBuchbar(){
        return plaetze>belegt;
    }
    public void einsteigen(){
        belegt ++;
    }
    public void aussteigen(){
        belegt--;
    }
    public String toString(){
        return bez+"  "+(plaetze-belegt);
    }
}

public class Bus extends Personentransport {
    private double dauer;
    private static int count = 1;


    public Bus(double dauer, int plaetze){
        super("Bus"+(count++), plaetze);
        this.dauer = dauer;
    }

    public double gesamtdauer(int km) {
        return dauer *km;
    }
}
```

Bewertun:

pro Fehler bzw nicht erfüllter Bedingung aus Aufgabenstellung 1-2 P Abzug

Aufgabe 8 (10 P)
Betrachten Sie die drei untenstehenden Klassendiagramme. Ist dadurch eine gültige (erlaubte) Java-Klassenstruktur gegeben?
Wenn nein, erläutern Sie kurz (!), warum nicht.
Wenn ja, geben Sie einen entsprechenden Java-Code an, der diese Struktur widerspiegelt.

Lösung 8
Das zweite Diagramm ist nicht korrekt, denn Klasse Z kann nicht Subklasse von X und Y sein.

Die anderen beiden Diagramme sind korrekt. Entsprechder Code:

```
public Interface A {}

public Interface B extends A { }

public class C implements A {}

public class D extends C implements B {}

public Interface M {}

public Interface L {}

public class K implements L, M {}
```

Bewertung:

pro Fehler 1P Abzug