**KLAUSUR 1 SAE E1FS 21.11.2017 Zeit: 75 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):

→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
→ In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 3+4+4 P

a) Welche 3 Möglichkeiten (3 Worte nennen) gibt es, einen Algorithmus darzustellen?
b) Erklären Sie die Begriffe Syntax und Semantik?
c) Was ist ein Compiler ?

2) 5 P

a) Wie viele Zustände genau kann man mit 20 Bit angeben? (als Hochzahl schreiben)
b) Welche der folgenden hintereinander ausgeführten Java-Anweisungen sind syntaktisch korrekt, welche nicht?
Falls diese syntaktisch korrekt sind, bitte jeweils die Werte der betreffenden Variablen nach Ausführung der jeweiligen Anweisung angeben:

```
...
y+2=x;
y=1+2;
y=y*y;
x=y*y;
...
```

3) 12P

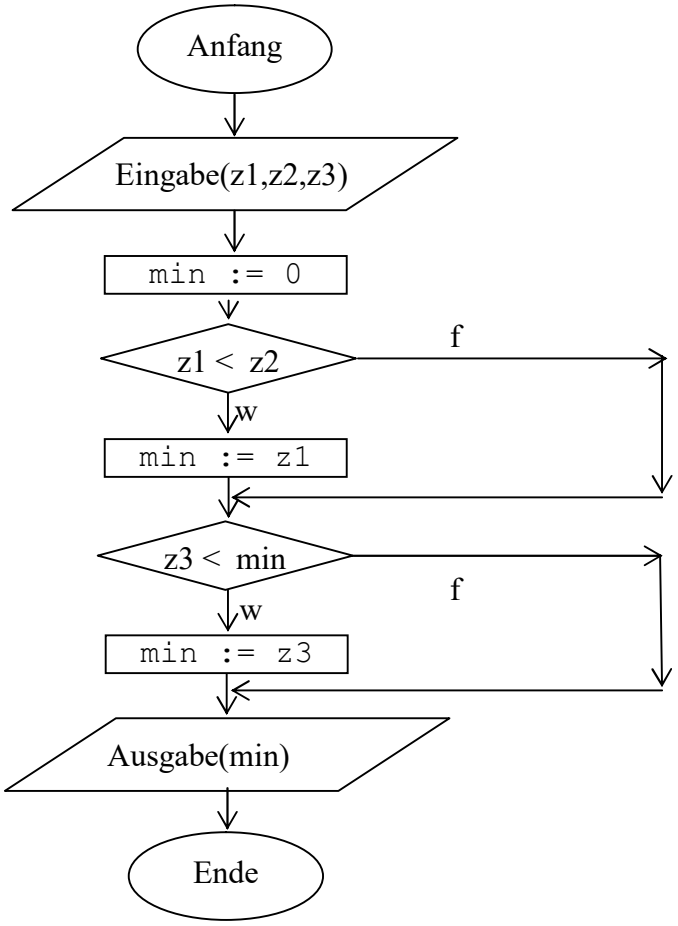
Erstellen Sie ein Flußdiagramm, das das Maximum dreier Zahlen berechnet und auf dem Bildschirm ausgibt.
EVA-Prinzip beachten!

4) 10P

Der folgende syntaktisch korrekte Teil eines Flußdiagramm soll die kleinste Zahl (dreier Zahlen z1, z2, z3) berechnen und in der Variablen min speichern.

Wenn das Programm korrekt ist, schreiben Sie "Der Algorithmus ist korrekt" und geben 5 Beispiele (mit konkreten Werten für z1, z2, z3 und dem berechneten Wert von min) an, wo er korrekt wird.

Wenn das Programm nicht korrekt ist, schreiben Sie " Der Algorithmus ist nicht korrekt" und geben ein **konkretes** Beispiel (mit konkreten Werten für z1, z2, z3 und dem berechneten Wert von min) an, bei dem der Algorithmus falsch wird.



5) 12P

Durch den folgenden Algorithmus (Flußdiagramm) soll von 50 Schülern jeweils der Prozentsatz der erreichten Punkte bzgl. der Gesamtpunktzahl in einer Klassenarbeit errechnet und ausgegeben werden.

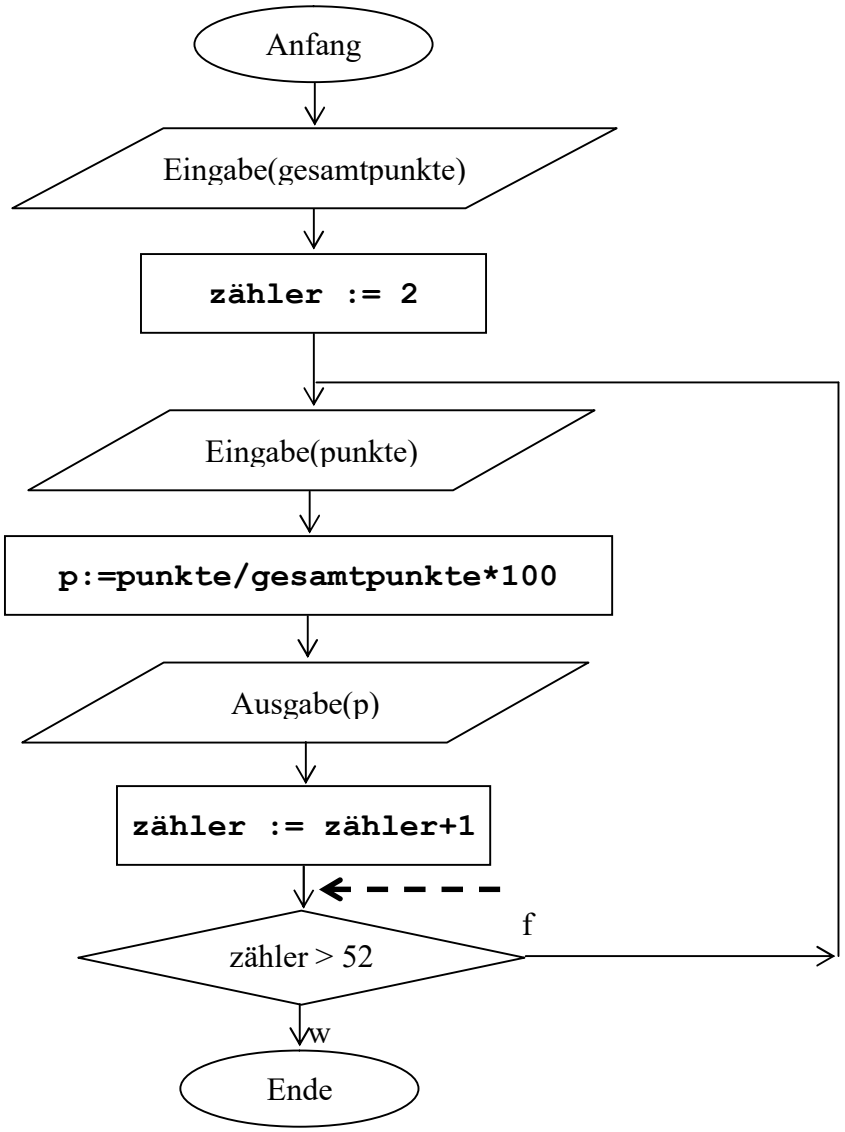
Dazu wird das Programm immer wieder an der mit <--- bezeichneten Stelle vor der Verzweigung (bei jedem Schleifendurchgang) gedanklich angehalten (Protokoll) und die Werte der Variaben zähler und die Anzahl der ausgegebenen Punkte der Schüler protokolliert.

a) Tragen Sie dazu in der Tabelle unten die Werte von zähler und die Anzahl der ausgegebenen Schüler bei den ersten 5 Durchgängen ein.

b) Geben Sie den Zusammenhang (Formel) zwischen zähler und Anzahl an.

Welche Werte haben zähler und Anzahl der ausgegebenen Schüler, wenn das Programm das **letzte** Mal an die mit <---bezeichnete Stelle kommt? c) Ist die folgende Lösung korrekt? Begründen Sie mit der Tabelle unten!

| Anzahl der ausgegebenen Schüler | Wert der Variablen zähler |
|---|---|
| | |
| | |
| | |
| | |
| | |



Lösungen:

1)

a) 3P

Flussdiagramm, Struktogramm, Programm

b) 4P

Die Syntax definiert die äußeren Formgesetze dieser Programmiersprache (ähnlich den grammatikalischen Regeln einer natürlichen - wie z.B. der englischen- Sprache).
Die Semantik ist der Bedeutungsinhalt (ähnlich der Bedeutung der einzelnen Worte einer natürlichen - wie z.B. der italienischen - Sprache) der einzelnen Objekte einer Programmiersprache.

c) 4P

Ein Compiler ist ein Übersetzer, der einen in einer höheren Programmiersprache formulierten Text (ein sogenanntes Programm) in einen aus Maschinenbefehlen bestehenden Text (einem sogenannten Maschinenprogramm) verwandelt. Dieses kann dann vom Mikroprozessor abgearbeitet (ausgeführt) werden. Außerdem überprüft er die Syntax des Programms.

2) 5 P

$2^{20} = (2^{10})^2 \approx (10^3)^2 = (10^3)^2 = 10^6$

```
y+2=x;    // syntaktisch falsch
y=1+2;    // y <-- 3
y=y*y;    // y <-- 9
x=y*y;    // x <-- 81
```

3) 10P

Der Algorithmus ist nicht korrekt:
z1 = 30 z2 = 20 z3 = 10 ==> min = 0

4) 12P

siehe Folie Präsentation

5) 12P

a)

| Anzahl der ausgegebenen Schüler | Wert der Variablen zähler |
|---|---|
| 1 | 3 |
| 2 | 4 |
| ... | ... |
| 51 | 53 |

b) anzahl = zähler - 2
zähler = 53, Anzahl der Schüler = 51
c) Programm nicht korrekt, da laut Tabelle 51 Schüler verarbeitet werden.

**KLAUSUR 1 SAE E1FI Nachtermin Zeit: 90 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):
→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Programme dürfen beim Kompilieren keine Warnungen und keine Fehlermeldungen bringen.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 3P

Geben Sie die 3 Eigenschaften eines Algorithmus an, die im Unterricht angesprochen wurden.

2) 5P

a) Wie viele Zustände genau kann man mit 8 Byte angeben? (als Hochzahl schreiben)
b) Näherungsweise Berechnung!

3) 6P

Der folgende Programmausschnitt veranlaßt, daß in den Variablen x und y die von einem Anwender eingegebenen Zahlen gespeichert werden.
Ergänzen Sie den Programmausschnit so, daß die Wert von x und y danach vertauscht werden.
Beispiel:
Nach Eingabe der Zahlenwerte haben x und y folgende Werte: x = 10 y=20
Nach der Ergänzung des Quellcodes (also kurz vor Programmende): x=20 y=10
Dies soll natürlich nicht nur für dieses Beispiel gelten, sondern für beliebige vom Anwender eingegeben Zahlen!

```
...
int x;
int y;

eingabe(x);
eingabe(y);
```

4) 12P

a)
Erstellen Sie ein Flußdiagramm, das das Produkt zweier ganzen Zahlen a und b berechnet, die alle größer oder gleich 0 sind. Es darf dabei nur die Addition + benutzt werden, keine Multiplikation.
Tipps: (mit einer Schleife und einem Zähler)
b=0 a*b = 0
b=1 a*b = a
b=2 a*b = a+a
b=3 a*b = a+a+a
b=4 a*b = a+a+a+a

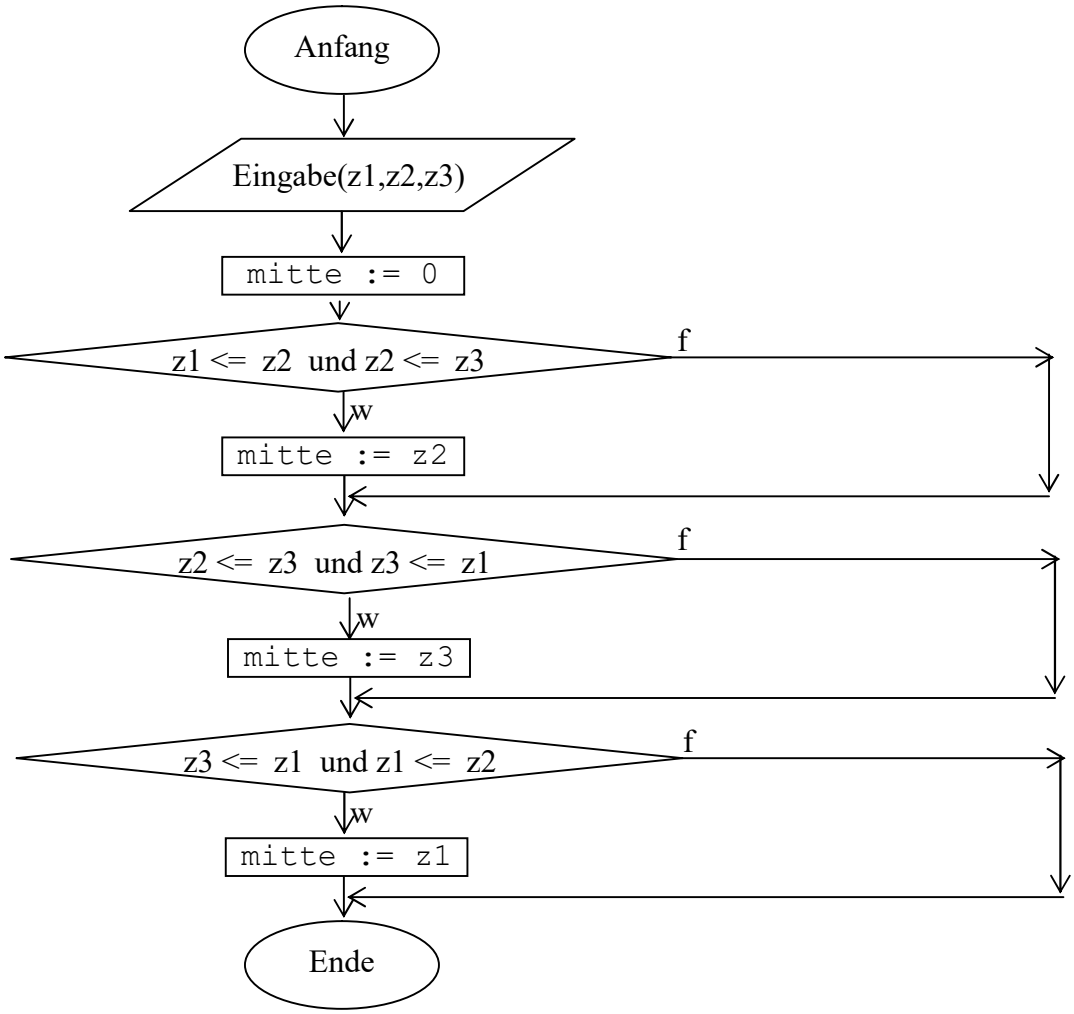
b)
Testen Sie ihr Flußdiagramm für folgende Fälle, d.h. tragen Sie jeweils die Werte für die Variable produkt in die folgende Tabelle ein:

| a | b | produkt |
|---|---|---|
| 5 | 0 | |
| 5 | 1 | |
| 5 | 2 | |
| 5 | 3 | |

5) 12 P

Der folgende syntaktisch korrekte Teil eines Flußdiagramms soll die mittlere Zahl (dreier Zahlen z1, z2. z3) berechnen und in der Variablen mitte speichern.
Wenn das Programm korrekt ist, schreiben Sie "Der Algorithmus ist korrekt" und geben 5 Beispiele (mit konkreten Werten für z1, z2, z3 und dem berechneten Wert von mitte) an, wo er korrekt wird.
Wenn das Programm nicht korrekt ist, schreiben Sie " Der Algorithmus ist nicht korrekt" und geben ein **konkretes** Beispiel (mit konkreten Werten für z1, z2, z3 und dem berechneten Wert von mitte) an, bei dem der Algorithmus falsch wird.
Beispiele für mitte

| z1 | z2 | z3 | mitte |
|---|---|---|---|
| 1 | 2 | 3 | 2 |
| 1 | 1 | -5 | 1 |
| 1 | 1 | 1 | 1 |
| 3 | 3 | 1 | 3 |



6) 12P

Durch den folgenden Algorithmus (Flußdiagramm) soll die Summe
3 + 4 + 5 + 6 + ... + 100 berechnet werden.
Dazu wird das Programm immer wieder an der mit <--- bezeichneten Stelle vor der Verzweigung (bei jedem Schleifendurchgang) gedanklich angehalten (Protokoll) und die Werte der Variaben i und sum protokolliert.

a) Tragen Sie dazu in der Tabelle unten die Werte von i und sum bei den ersten 5 Durchgängen ein. Bitte sum **nicht** berechnen, sondern die Summe jeweils darstellen, wie z.B. 7 + 9 + 11.

b) Welche Werte haben i und sum, wenn das Programm das **letzte** Mal an die mit <--- bezeichnete Stelle kommt?

c) Ist das Programm semantisch korrekt (d.h. wird also durch das Programm die Summe 3 + 4 + 5 + 6 + ... + 100 berechnet). Bitte Begründung angeben!

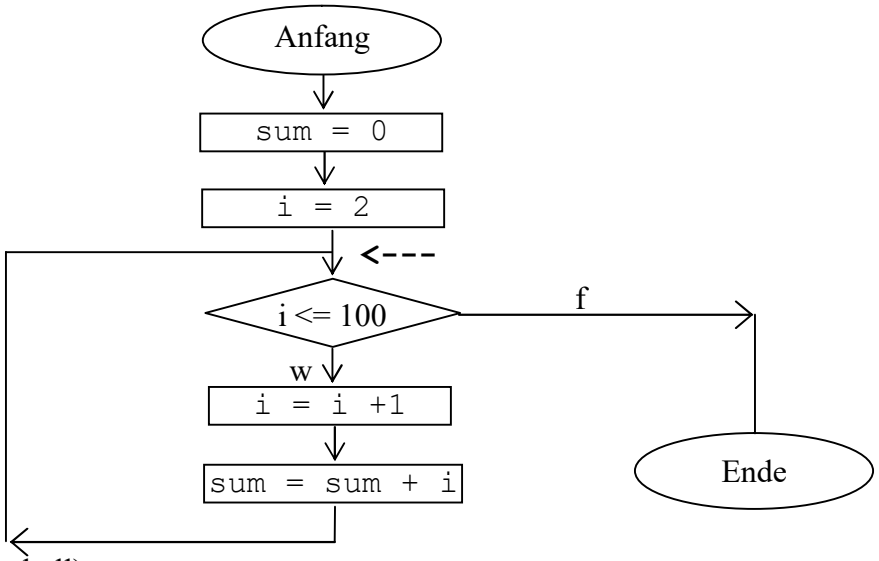



Tabelle (Protokoll):

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| i | | | | | | | |
| sum | | | | | | | |

Lösungen:

1) 3P

eindeutig, schrittweise, endlich

2) 5 P

a) $2^{64}$

b) $2^{64} = 2^{60+4} = 2^{60} \cdot 2^4 = 16 \cdot 2^{60} = 16 \cdot 2^{10 \cdot 6} = 16 \cdot (2^{10})^6 \approx 16 \cdot (10^3)^6 = 16 \cdot 10^{18}$

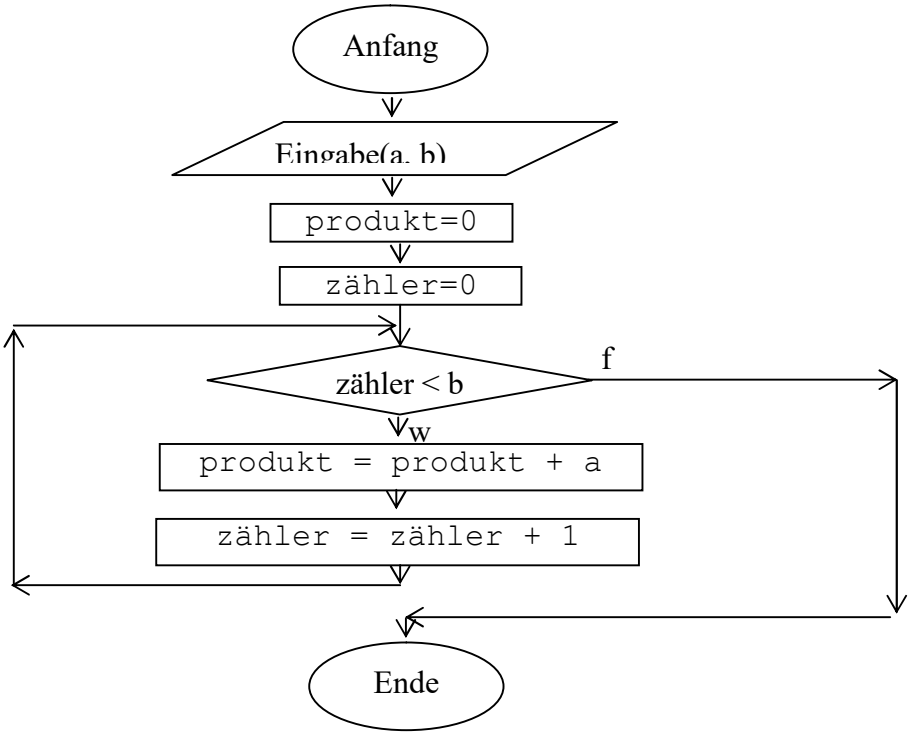
3) 6P

```
int x;
int y;
int speichereX;
int speichereY;

eingabe(x);
eingabe(y);
speichereX=x;
speichereY=y;
x= speichereY;
y= speichereX;
```

4) 12P

a)




b)

| a | b | produkt |
|---|---|---|
| 5 | 0 | 0 |
| 5 | 1 | 5 |
| 5 | 2 | 5+5 |
| 5 | 3 | 5+5+5 |

5) 12P

Algorithmus falsch für
z1=1, z2=3 , z3=2

6) 12P

a) 5P

| i | 2 | 3 | 4 | 5 | 6 | ... | 101 |
|---|---|---|---|---|---|---|---|
| sum | 0 | 0+3=3 | 3+4 | 3+4+5 | 3+4+5+6 | ... | 3+...+101 |

b) 4P

i = 101 und sum = 3+...+101

c) 3P

Da nach das Programm an dieser Stelle beendet wird, haben i und sum die bei b) protokollierten Werte. Das Programm ist also nicht korrekt.

**KLAUSUR 2 SAE E1FS 19.12.2017 Zeit: 75 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):
- → Alle Aufgaben müssen bearbeitet werden.
- → Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- → Aufgabenblätter bitte auch abgeben.
- → Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- → Die rote Farbe darf nicht benutzt werden.
- → Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- → Selbsterklärende Variablennamen benutzen.
- → Programme müssen benutzerfreundlich sein.
- → EVA-Prinzip muss benutzt werden.
- → Einrücken der entsprechenden Programmteile.
- → Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
- → In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
- → Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 5 P

Erstellen Sie ein Flußdiagramm, das eine Endlosschleife enthält.

2) 5 P

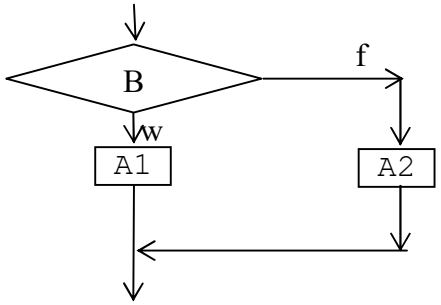
Welchen Wert besitzt im folgenden Programmauschnitt die Variable w nach Ausführung der mit (*) bezeichneten Anweisung?
Begründen Sie!

```
...
double w;
int i = 2;
w = 10/((int)50.3 * i);     // (*)
...
```

3) 5 P

Simulieren Sie den folgenden Ausschnitt eines Flußdiagramms durch ein Flußdiagramm, in dem nur einseitige Verzweigungen vorkommen.

Tipp: Die Bedingung „nicht B“ können Sie mit „non B“ bezeichnen




4) 5 P

Falls der folgende Programmauschnitt Fehler besitzt, geben Sie die Zeilennummern an, wo sich der Fehler befindet.
Falls er keine Fehler besitzt, schreiben Sie „fehlerfrei“.

```
...
int i = 3.0;        // (1)
double d = 4.5;     // (2)
i = i *d;           // (3)
...
```

5) 5 P

Der folgende syntaktisch korrekte Programmausschnitt soll das Maximum (zweier über Tastatur eingegebener Zahlen z1, z2) berechnen und in der Variablen max speichern.
(Die Eingabe wird hier durch eingabe(z1,z2) abgekürzt),
Wenn das Programm korrekt ist, schreiben Sie "Der Algorithmus ist korrekt" und geben 2 Beispiele (mit konkreten Werten für z1, z2 und dem berechneten Wert von max) an, wo er korrekt wird.
Wenn das Programm nicht korrekt ist, schreiben Sie " Der Algorithmus ist nicht korrekt" und geben ein **konkretes** Beispiel (mit konkreten Werten für z1, z2 und dem berechneten Wert von max) an, bei dem der Algorithmus falsch wird.

```
…
int z1;
int z2;
int max=0;
eingabe(z1,z2);
if(z1<z2){
  max=z2;
}
if(z2<z1){
  max=z1;
}
…
```

6) 5 P

Erstellen Sie ein Flußdiagramm, das 3 über Tastatur eingegebene Zahlen (gespeichert in den Variablen a, b,c) wie folgt vertauscht:
a -> b -> c -> a
Beispiel:

| a | b | c | ==> | a | b | c |
|---|---|---|---|---|---|---|
| 3 | 1 | 7 | | 7 | 3 | 1 |
| 1 | 2 | 3 | | 3 | 1 | 2 |

7) 6 P

Erstellen Sie ein Flußdiagramm, das von 3 über Tastatur eingegebenen Zahlen (gespeichert in den Variablen a, b,c) die Anzahl der Zahlen (von diesen 3 Zahlen) berechnet (und in der Variablen anzahl speichert), die kleiner als 0 sind.

8) 7 P

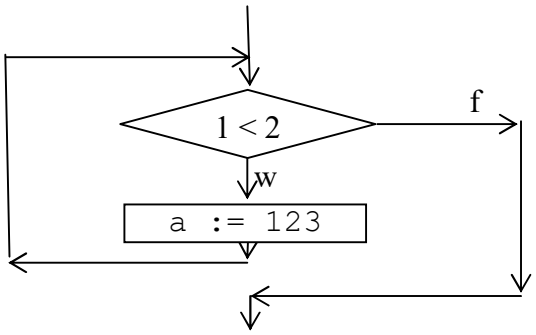
Erstellen Sie ein Flußdiagramm, das die folgende Summe berechnet und in der Variablen summe abspeichert:
summe = 100 + 101 + 102 + …+ 9999 + 10000

9) 8 P

Erstellen Sie ein Flußdiagramm, das von 2 über Tastatur eingegebenen ganzen Zahlen (die jeweils größer als 0 vorausgesetzt werden dürfen und in den Variablen z und t gespeichert werden) berechnet, wie oft t in z enthalten ist. Dieser Wert muß in der Variablen anzahl gespeichert werden.
Der Operator / darf dabei nicht benutzt werden.
Beispiele:
t=40 ist 2 Mal in z=90 enthalten, weil 90-40-40=10, also anzahl = 2
t=20 ist 3 Mal in z=60 enthalten, weil 60-20-20-20=0, also anzahl = 3
t=6 ist 5 Mal in z=34 enthalten, weil 34-6-6-6-6-6=4, also anzahl = 5

Lösungen
Bemerkung:
Bei manchen Flussdiagrammen wurde - mangelnden Platzes wegen - die Bezeichner "Anfang" bzw. "Ende" weggelassen.

1) 5 P

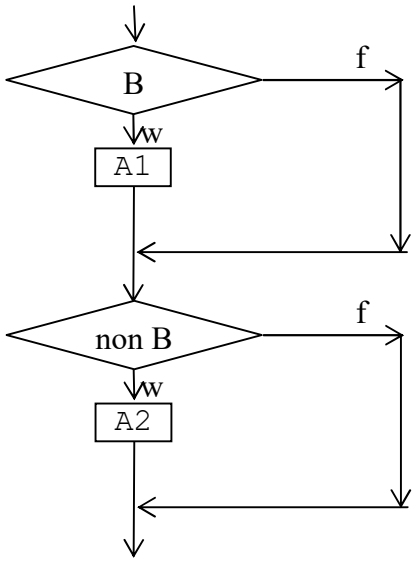



2) 5 P

```
…
double w;
int i = 2;
w = 10/((int)50.3 * i);
…
```

w = 0
10 / 100 entspricht integer / integer. Dabei wird der Nachkommateil abgeschnitten.

3) 5 P



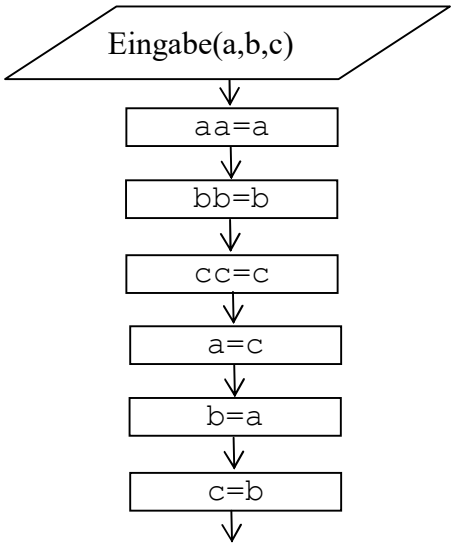

4) 5 P

Zeile 1: double kann nicht in int abgespeichert werden
Zeile 3: double kann nicht in int abgespeichert werden

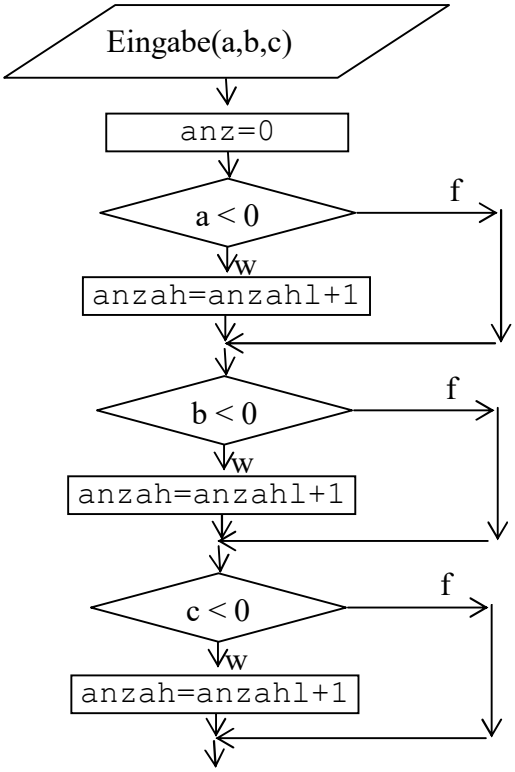
5) 5 P

z1 = 20, z2 = 20 --> max = 0

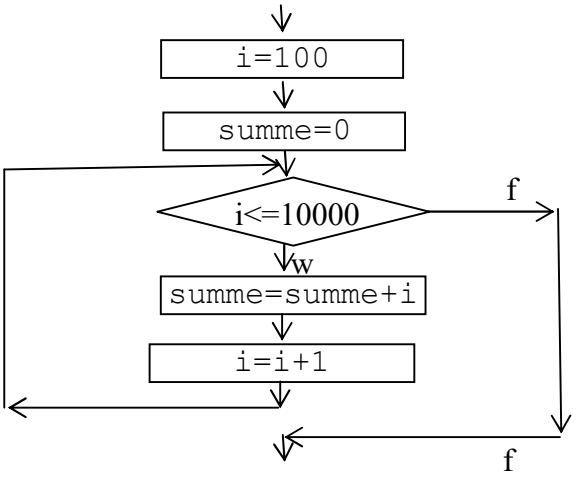
6) 5 P


Eingabe(a,b,c)
aa=a
bb=b
cc=c
a=c
b=a
c=b

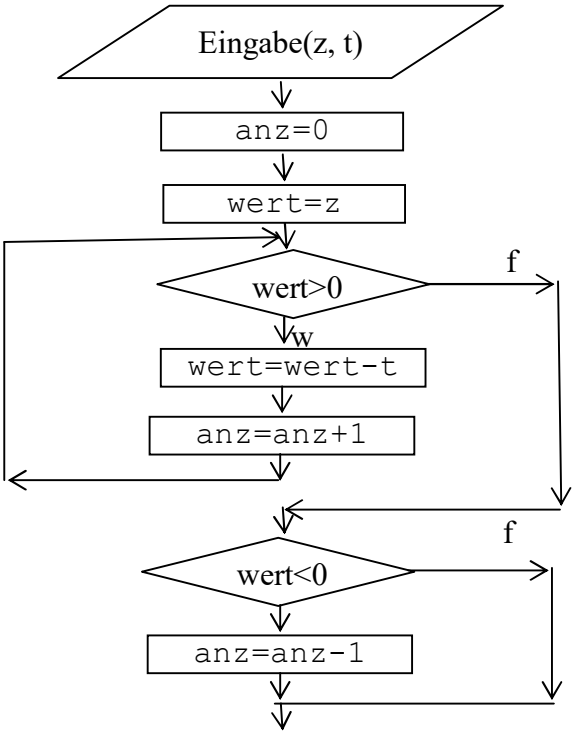
7) 6 P


Eingabe(a,b,c)
anz=0
a < 0
f
w
anzah=anzahl+1
b < 0
f
w
anzah=anzahl+1
c < 0
f
w
anzah=anzahl+1

8) 7 P


i=100
summe=0
i<=10000
f
w
summe=summe+i
i=i+1
f

9) 8 P


Eingabe(z, t)
anz=0
wert=z
wert>0
f
w
wert=wert-t
anz=anz+1
f
wert<0
anz=anz-1

**KLAUSUR 2 SAE E1FI Nachtermin Zeit: 75 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):

→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Programme dürfen beim Kompilieren keine Warnungen und keine Fehlermeldungen bringen.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 25P

Eine natürliche Zahl n (n>=2) heißt Primzahl, wenn sie nur durch 1 und sich selbst teilbar ist.
Beispiele für Primzahlen: 2, 3, 5, 7, 11, 13, ....

a) Erstellen Sie ein Flussdiagramm, das folgendes macht:
Es muß von einer eingegebenen ganzen Zahl z >= 2 feststellt werden, ob diese eine Primzahl ist. Das Ergebnis muß dann auf dem Bildschirm ausgegeben werden.

b) Erstellen Sie dazu ein Testprotokoll mit 5 verschiedenen Tests.

Tipp: Dividieren Sie die zu überprüfende Zahl z durch 1, 2, 3, 4, .., z und zählen wie oft "teilt" erfolgreich ist (siehe unten).

2) 25P

Erstellen Sie ein Flussdiagramm, in dem zwei ganze (>0) Zahlen (Zähler und Nenner) über Tastatur eingegeben werden und in dem Zähler und Nenner maximal gekürzt werden.
Beispiel: 18 / 24 = 3 / 4

Bemerkung:
Sie dürfen den Operator "teilt" verwenden, der auf ganze Zahlen angewendet werden kann.
Falls x und y ganze Zahlen sind: x teilt y
erg = x teilt y
erg ist true, falls die Zahl x die Zahl y teilt.

Beispiel:

```
erg = 3 teilt 9;     // erg hat den Wert true
erg = 4 teilt 11;    // erg hat den Wert false
x = 2;
y = 10;
erg = x teilt y;     // erg hat den Wert true.
```

**KLAUSUR 3 SAE E1FS 17.4.2018 Zeit: 75 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):

→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
→ In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 6P

Geben Sie 2 Unterschiede zwischen einer While-Anweisung und einer do-while-Anweisung an.
Geben Sie ein **konkretes** Beispiel (Programmteil) an, bei dem bei **gleicher** Schleifenbedingung und **gleichem** Schleifenrumpf eine do-while-Anweisung semantisch etwas anderes macht, als eine while-Anweisung.

2) 4P

Gegeben sind folgende syntaktisch korrekte Programmteile:

a)
```
i=10;
while(i<20)
  System.out.println("Hallo Welt\n");
```

b)
```
i=20;
while(i<20)
  System.out.println("Hallo Welt\n");
```

Wie oft gibt der Programmteil bei a) bzw. b) die Meldung "Hallo Welt" auf dem Bildschirm aus ? Begründen Sie !

3) 6P

In den folgenden Programmauschnitten gibt der Anwender für i eine ganze Zahl ein.
a) Für welche Werte von i bringen die Programmausschnitte die gleiche Anzahl von Meldungen auf den Bildschirm ?
b) Für welche Werte von i bringen die Programmausschnitte nicht die gleiche Anzahl von Meldungen auf den Bildschirm ?

```
eingabe(i);
while (i<=3){
  i = i+1;
  Sop("Hallo Welt");
}
```

```
eingabe(i);
do{
  i = i+1;
  Sop("Hallo Welt");
} while(i<=3);
```

Bem: Sop ist die Abkürzung für `System.out.println`

4) 10P

a) Ein Anwender soll eine ganze Zahl zwischen 1 und 5 (je einschließlich) über Tastatur eingeben. Schreiben Sie dazu einen Programmausschnitt in Java, in dem der Anwender gezwungen wird. so lange eine Zahl einzugeben, bis diese Bedingung erfüllt ist. Erst dann soll im Programm die nächste Anweisung erreicht werden.
Die Eingabe einer ganzen Zahl kann man mit dem folgenden Pseudocode schreiben (wobei z eine integer-Variable bedeutet): eingabe(z);

b) Geben Sie den Programmverlauf in einem Schreibtischtest (Protokoll mit Belegung der Variablen) an für:
b1) Anwender gibt Zahl 3 ein und b2) Anwender gibt Zahl 123 ein.

5) 10P

a) Geben Sie einen Programmausschnitt in Java an, der mit Hilfe einer do-while-Schleife die folgende Summe berechnet:
13 + 15 + 17 + 19 + ... + 2047
Es wird empfohlen einen Schreibtischtest (Protokoll mit Belegung der Variablen) zu machen. Bei fehlendem Schreibtischtest und falscher Lösung gibt es 0 Punkte.

6) 15P

Erstellen Sie ein Struktogramm eines Programms, das die Potenz $a^n$ einer reellen Zahl a bestimmt.
n muß eine **ganze** Zahl sein.
Im Hauptprogramm muss der Anwender a und eine ganze Zahl n eingeben können.
Das Programm berechnet dann das Ergebnis $a^n$ und gibt es auf dem Bildschrim aus.
Wichtig: Das Ergebnis muß in der Variable erg gespeichert werden!

Definition der Potenz:

$a^n = \underbrace{a \cdot a \cdot ... \cdot a}_{\text{n - mal}}$, wenn n > 0

$a^n = 1$, wenn n = 0

$a^{-n} = \frac{1}{a^n}$, wenn n <0

Beispiele:

$2^4 = 2 \cdot 2 \cdot 2 \cdot 2 = 16$

$5^0 = 1$

$3^{-2} = \frac{1}{3^2} = \frac{1}{9}$

Lösungen:

1) 6P

Anzahl Durchgänge der do-while-Anweisung: >= 1
Anzahl Durchgänge der while-Anweisung: >= 0
while-Anweisung: vor dem Schleifenrumpf wird Bedingung überprüft
do-while-Anweisung: nach dem Schleifenrumpf wird Bedingung überprüft

```
i=10;
do{
  System.out.println(i);
}
while(i<10);
```

```
i=10;
while(i<10){
  System.out.println (i);
}
```

2) 4P

a) Unendlich oft, weil die Bedingung immer wahr ist.
b) Null Mal, weil die Bedingung falsch ist.

3) 6P

i< = 3: gleiche Anzahl von Meldungen
i > 3 : verschiedene Anzahl von Meldungen

4) 10P

a) 6P

```
...
do {
  System.out.println("ganze Zahl zwischen 1 und 5 eingeben");
  eingabe(i);
}while(!(i>=1 && i<=5));
```

b) 4P

Eingabe: i = 3
(!(3>=1 && 3<=5)) wird falsch und damit wird Schleife verlassen
Eingabe: i = 123
(!(123>=1 && 123<=5)) wird wahr und damit wird Schleife wiederholt.

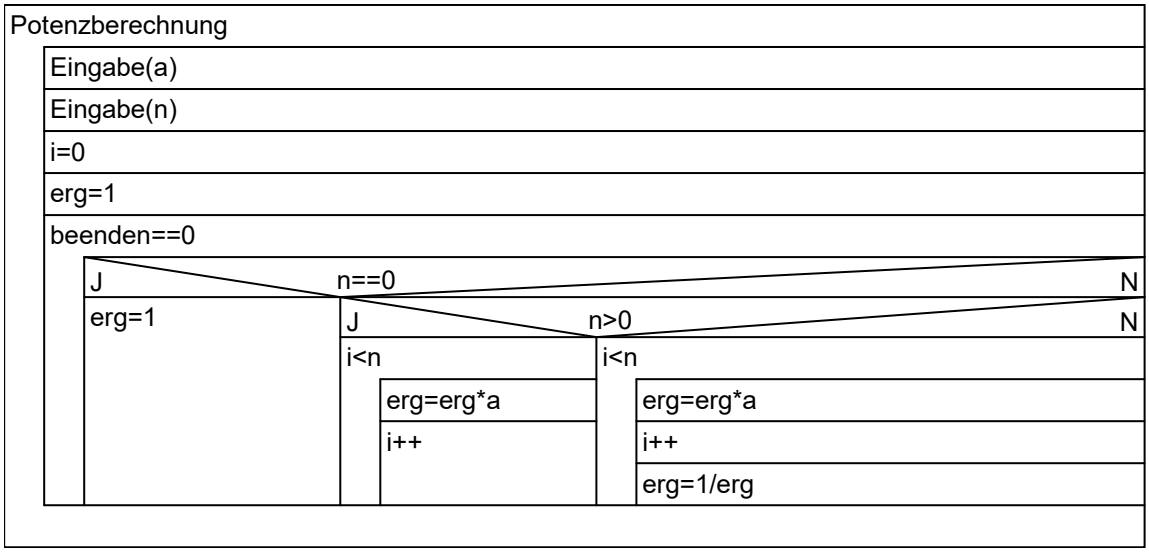
5) 10P

a) 10P

```
...
int summe=0;
int zahl=13;
do{
  summe = summe + zahl;
  zahl = zahl +2;
} while (zahl <=2047);

...
```

6) 15P


Potenzberechnung
Eingabe(a)
Eingabe(n)
i=0
erg=1
beenden==0
J
n==0
N
erg=1
J
n>0
N
i<n
i<n
erg=erg*a
i++
erg=erg*a
i++
erg=1/erg

**KLAUSUR 3 SAE E1FI Nachtermin1 Zeit: 75 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):

→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Programme dürfen beim Kompilieren keine Warnungen und keine Fehlermeldungen bringen.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

# AUFGABEN

1) 25P

Herr X behauptet: Wenn du das Folgende machst, kommt immer die gleiche Zahl heraus: Schreibe eine positive dreistellige Zahl auf ein Papier. Schreibe rechts davon die gleiche Zahl nochmals auf das Papier. Dies ergibt eine sechsstellige Zahl.
Wenn man diese durch 7 dividiert, kommt immer als Rest 0 heraus.
a) Schreiben Sie ein Programm, das diese Behauptung für eine über Tastatur eingegebene Zahl überprüft, d.h. den Rest ausgibt.
b) Schreiben Sie ein Java-Programm, das diese Behauptung für alle dreistellige Zahlen überprüft.

2) 25P

Entwickeln Sie ein Java-Programm für einen Tilgungsplan:
Nach Angabe von Kreditsumme, Darlehnszinsen pro Jahr (umrechnen in Darlehnszinsen pro Monat, siehe Rechenbeispiel unten) und monatlicher Rate sollen die Dauer der Tilgung und die Gesamtkosten (Summe aller Zinsen, die an die Bank gezahlt wurden !) berechnet werden. Die monatliche Rate wird zur Zinszahlung und Tilgung benutzt.

Mathematische Anleitung anhand eines Beispiels:
Voraussetzungen: Kreditsumme S = 1000 EURO, Darlehnszinsen pro Jahr p = 120%, d.h. 120/12 = 10% Darlehnszinsen pro Monat, monatliche Rate = 500 EURO.

| n | Zinsen $z_n$ | Tilgung $tg_n$ | Schulden $S_n$ |
|---|---|---|---|
| 0 | 0 | 0 | 1000 |
| 1 | 1000 · 0,1 = 100 | 500 - 100 = 400 | 1000 - 400 = 600 |
| 2 | 600 · 0,1 = 60 | 500 - 60 = 440 | 600 - 440 = 160 |
| 3 | 160 · 0,1 = 16 | 160 | 0 |

Im 3. Monat tilgt der Kunde 160 Euro und zahlt noch 16 Euro Zinsen. Deshalb bekommt er noch 500 - (16 + 160) = 324 Euro zurück.
Insgesamt hat man an die Bank 100 + 60 + 16 = 176 Euro Zinsen gezahlt.

Bem: Tastatureingaben können mit eingabe(z) abgekürzt werden, wobei z eine Variable ist, in der die Eingabe gespeichert wird.

**KLAUSUR 3 SAE E1FI Nachtermin2 Zeit: 75 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):
- → Alle Aufgaben müssen bearbeitet werden.
- → Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
- → Aufgabenblätter bitte auch abgeben.
- → Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
- → Die rote Farbe darf nicht benutzt werden.
- → Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
- → Selbsterklärende Variablennamen benutzen.
- → Programme müssen benutzerfreundlich sein.
- → EVA-Prinzip muss benutzt werden.
- → Einrücken der entsprechenden Programmteile.
- → Programme dürfen beim Kompilieren keine Warnungen und keine Fehlermeldungen bringen.
- → Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 25P

Schriftlich Wurzel ziehen

Um die Wurzel einer positiven Zahl z zu berechnen, sucht man zwei Zahlen z1 und z2, deren Produkt die Zahl z ergibt. Zusätzlich muß aber noch die Nebenbedingug z1 = z2 gelten!

Beispiel: Berechne $\sqrt{3}$

Wenn einem nichts besseres einfällt, setzen wir z1 = 1, dann muß z2 = z/z1 = 3/1 = 3 sein, also:

z1 := 1

z2 := z / z1 = 3 / 1 = 3

Da z1 und z2 noch meilenweit voneinander entfernt sind (also $z1 \neq z2$ gilt), müssen wir wenigstens versuchen, z1 und z2 sich einander näher zu bringen.

Wir starten also einen neuen Versuch für die Bestimmung von z1 und z2.

Wir nehmen jetzt für z1 den Mittelwert von dem alten z1=1 und dem alten z2=3.

also:

z1 := (z1+z2) / 2 = (1+3) / 2 = 2

z2 := z / z1 = 3 / 2 = 1,5

z1 und z2 sind jetzt nicht mehr so stark voneinander entfernt wie vorher, sind aber trotzdem noch voneinander entfernt. Deshalb machen wir weitere Versuche, um sie einander näher zu bringen:

z1 := (z1+z2) / 2 = (2+1,5) / 2 = 1,75

z2 := z / z1 = 3 / 1,75 = 12/7 ≈ 1,7143

Man kann die Berechnungen für z1 und z2 ewig weitermachen, trotzdem werden z1 und z2 nie gleich groß werden. Das macht aber nichts aus, denn wir machen das dann eben so lange weiter, bis sich z1 und z2 nur noch betragsmäßig um einen vorgegeben kleinen Wert, wie z.B. $\varepsilon = 10^{-5}$ unterscheiden. Dann sind wir fertig!

Schreiben Sie dazu ein passendes Java-Programm.

2) 25P

Münze werfen bis jede Zahl erreicht wird

Ein Mathematiker behauptet, daß man eine Münze im Durchschnitt 3 Mal werfen muss, bis jede der zwei Seiten mindestens einmal erschienen ist.

Simulieren Sie diese Aufgabe durch ein entsprechendes Java-Programm.

Für die Berechnung eines Münzwurs benutzen Sie die Funktion

wurf = zufallMünze();

Die variable Wurf bekommt dann den Wert 0 oder 1.

**KLAUSUR 4 SAE E1FS 26.6.2018 Zeit: 45 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):

→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß in DRUCKSCHRIFT auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.
→ In allen Klassen dürfen in keiner Methode Ein- oder Ausgaben (auf dem Bildschirm) vorkommen, außer es handelt sich um reine Ausgabe- oder Eingabe Methoden.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 7P

gegeben ist der folgende, syntaktisch korrekte Programmausschnitt (Codeschnipsel):

```
...
double a = Math.PI;
int i;
for(i=1;i<4;i++){
  a = a * a;
}
System.out.println("a="+a);
...
```

Was wird auf dem Bildschirm ausgegeben ? Geben Sie den **genauen** Wert von a an!
Begründen Sie, indem Sie den Wert von a bei jedem Schleifendurchgang angeben!

Bemerkung:
a) Vor der Schleife hat a den Wert der Kreiszahl π.
b) Der genaue Wert von π * π ist $\pi^2$ , nicht 9,86902225 !!!

2) 14P

a) Was ist eine Klasse ?
b) Was ist ein Objekt?
c) Was enthält eine Klasse?
d) Was ist der Sinn einer Methode?
e) Was ist ein Attribut?
f) Was bedeutet private?
g) Was bedeutet public?

3) (Alle Teilaufgaben müssen in **einem** Programm realisiert werden) 30P

**Sichtbarkeits- und Zugriffsmodifizierer wie public, usw. dürfen nicht weggelassen werden.**

a)

Erstellen Sie die Klasse Hund, (mit genau den 2 Attributen "name" und "alter", genau 2 set-Methoden, genau 2 get-Methoden und genau 2 Konstruktoren).

b)

Es muss ein 5 Jahre alter Hund mit dem Namen "RRRex" erzeugt werden.
Es muss ein 7 Jahre alter Hund mit dem Namen "Fasss" erzeugt werden.
Erzeugen Sie die dazugehörige Java-Anweisungen.

c)

RRRex wird 9 Jahre alt. Erzeugen Sie die dazugehörige Java-Anweisung.
Fasss wird 10 Jahre alt. Erzeugen Sie die dazugehörige Java-Anweisung.

d)

Ein Hacker hat das Alter das Alter von RRRex auf einen bestimmten Wert gesetzt.
Geben Sie das Alter von RRRex durch die entsprechende Java-Anweisung auf dem Bildschirm aus.

e)

Ein Hacker hat das Alter von Fasss auf einen bestimmten Wert gesetzt.
Machen Sie danach Fasss um 2 Jahre älter.
Realisieren Sie das durch die entsprechende Java-Anweisung(en).

Lösungen

1) 7P

$\pi^8$

2) 14P

a) Ein Bauplan, nach dem ein Objekt erstellt wird
b) Ein nach einem Bauplan im Arbeitspeicher angelegter, reservierter Speicher.
c) Attribute und Methoden.
d) Attribute zu lesen und zu beschreiben.
e) Daten in einer Klasse.
f) Auf private Member darf nur innerhalb einer Klasse zugegriffen werden.
g) Auf public Member darf innerhalb und außerhalb einer Klasse zugegriffen werden.

3)

```
public class Startklasse {
    public static void main(String[] args) throws Exception {
        // b)    4P
        Hund myh1;
        Hund myh2;
        myh1 = new Hund("RRRex", 5);
        myh2 = new Hund("Fasss", 7);
        // c)    4P
        myh1.setAlter(9);
        myh2.setAlter(10);
        // d)    2P
        System.out.println("Alter= " + myh1.getAlter());
        // e)    4P
        myh2.setAlter(myh2.getAlter()+2);
    }
}

// a)    16P
class Hund {
    private String name;          // 1P
    private double alter;         // 1P

    public Hund() {               // 3P
        name = "Bello";
        alter = 0;
    }

    public Hund(String pName, double pAlter) {  // 3P
        name = pName;
        alter = pAlter;
    }

    public void setName(String n) {             // 2P
        this.name = n;
    }

    public void setAlter(double pAlter) {       // 2P
        alter = pAlter;
    }

    public String getName() {                   // 2P
        return (name);
    }
```

```
    public double getAlter() {                    // 2P
        return (alter);
    }

}
```

**KLAUSUR 4 SAE E1FI Nachtermin Zeit: 60 Minuten**

*Name, Vorname:*

Hilfsmitttel:
keine

Hinweise (unbedingt beachten):

→ Alle Aufgaben müssen bearbeitet werden.
→ Der Name und Vorname muß auf jedes Aufgabenblatt und auf jedes Lösungsblatt geschrieben werden.
→ Aufgabenblätter bitte auch abgeben.
→ Die Lösungsblätter müssen in folgender Form durchnummeriert werden. Beispiel: 1/4 2/4 3/4 4/4
→ Die rote Farbe darf nicht benutzt werden.
→ Lassen Sie bitte auf der linken Seite einen mindestens 3cm breiten Rand.
→ Selbsterklärende Variablennamen benutzen.
→ Programme müssen benutzerfreundlich sein.
→ EVA-Prinzip muss benutzt werden.
→ Einrücken der entsprechenden Programmteile.
→ Programme dürfen beim Kompilieren keine Warnungen und keine Fehlermeldungen bringen.
→ Bei Nichtbeachtung dieser Hinweise gibt es einen Punkteabzug !!!!

## AUFGABEN

1) 25 P

Eine natürliche Zahl n > 0 heißt vollkommene Zahl (perfekte Zahl), wenn sie gleich der Summe aller ihrer positiven Teiler (außer sich selbst) ist.
Beispiel:
Die Summe der Teiler vom 6 (außer sich selbst): 1 + 2 + 3 = 6, also ist 6 vollkommen.

Schreiben Sie ein Programm, das bestimmt, ob eine natürliche Zahl n > 0 vollkommen ist.

2) 25 P

Eine 6-stellige Zahl heißt "symmetrisch", wenn die Ziffern der ersten Hälfte der Zahl in umgekehrter Reihenfolge die Ziffern der zweiten Hälfte der Zahl bilden.
Beispiel: 396693.
Schreiben Sie ein Programm, das bestimmt, ob eine 6-stellige Zahl eine symmetrische Zahl ist: Sie geben über Tastatur eine 6-stellige Zahl n (n>=0) ein, wobei das Programm nachprüft, ob diese Zahl 6-stellig ist. Auf dem Bildschirm soll dann ausgegeben werden, ob diese Zahl eine symmetrische Zahl ist.