Programmieren 1		Name	
WS2x/2x		Semester	
Prof. Dr. S. Kruse		Matrikel-Nr.	
		Hilfsmittel	Code aus den Laborübungen
Prüfungsart	Klausur	*Max. Punkte*	100
Prüfungsdatum	xx, xx.xx.xx	*Punkte*	
Prüfungsdauer	90 min.	*Note*	

Hinweise:

- Die Klausur umfasst 5 Seiten mit 4 Aufgaben. Prüfen Sie bitte zuerst, ob Sie alle Blätter erhalten haben.
- Geben Sie auf allen Lösungsblättern Ihren Namen, das Semester und die Matrikelnummer an.
- Falls Sie zusätzliches Papier benötigen, melden Sie sich bitte.
- Schreiben Sie bitte deutlich, sauber und leserlich und mit einem dokumentenechten Schreibgerät, also nicht mit Bleistift.
- Benutzen Sie keine rote Farbe.
- Streichen Sie Ungültiges deutlich und eindeutig durch. Verwenden Sie kein Tipp-Ex, Korrekturband u. ä.
- Bieten Sie zu einer Fragestellung nicht mehrere Lösungen an, sondern entscheiden Sie sich für eine und streichen Sie die anderen durch.
- Lösungswege müssen nachvollziehbar sein. Die Angabe von Endergebnissen genügt nicht.
- Es sind keine `import`-Anweisungen erforderlich.

- Viel Erfolg!

Achtung: Die Klausur umfasst 105 Punkte und damit etwas mehr an Aufgaben als eine reguläre Klausur mit 100 Punkten.

1. Aufgabe: Allgemeine Fragen (35 Punkte)

1. Schreiben Sie die Umrechnungsschritte nieder, um die Dezimalzahl `51` in eine Dualzahl, Oktalzahl und Hexadezimalzahl zu wandeln.

2. Markieren und erläutern Sie eventuelle Fehler bei den folgenden Deklarationen und Definitionen:

   (a) `double long_way_to_go = 0,0;`

   (b) `bool $bval;`

   (c) `int 32bitint;`

3. Werten Sie den folgenden Ausdruck schrittweise wie ein Computer aus:

$$(17 - 4) * ((8 - 2)/3 + 6)$$

Fortsetzung Aufgabe 2:

4. Wandeln Sie den Ausdruck $x_1 \wedge (x_1 \Rightarrow x_2) \wedge (x_2 \Rightarrow x_3) \wedge \overline{x_3}$ in einen boolschen Ausdruck um und vereinfachen Sie diesen danach mit Hilfe der unten angegebenen aussagenlogischen Sätze.

1. $a \wedge b \equiv b \wedge a$
2. $a \vee b \equiv b \vee a$
3. $(a \wedge b) \wedge c \equiv a \wedge (b \wedge c)$
4. $(a \vee b) \vee c \equiv a \vee (b \vee c)$
5. $(a \vee b) \wedge c \equiv (a \wedge c) \vee (b \wedge c)$
6. $(a \wedge b) \vee c \equiv (a \vee c) \wedge (b \vee c)$
7. $a \wedge 1 \equiv a$
8. $a \vee 0 \equiv a$
9. $a \wedge \bar{a} \equiv 0$
10. $a \vee \bar{a} \equiv 1$
11. $a \wedge a \equiv a$
12. $a \vee a \equiv a$
13. $a \wedge 0 \equiv 0$
14. $a \vee 1 \equiv 1$
15. $a \vee (a \wedge b) \equiv a$
16. $a \wedge (a \vee b) \equiv a$
17. $a \vee (\bar{a} \wedge b) \equiv a \vee b$
18. $a \wedge (\bar{a} \vee b) \equiv a \wedge b$
19. $\bar{\bar{a}} \equiv a$
20. $\bar{a} \wedge \bar{b} \equiv \overline{(a \vee b)}$
21. $\bar{a} \vee \bar{b} \equiv \overline{(a \wedge b)}$

   Wie lässt sich der Ausdruck interpretieren?

5. Gegeben sei folgender Code:

```
if (a-b < 10 && -10 < a-b) {
  c = false;
}
```

   Formen Sie den Code in äquivalenten Code um, der ohne boolsche Operation auskommt.

6. Gegeben seien folgende Pfade: $A \to B \to C$, $A \to D$, $A \to E$, $A \to E \to F$, $E \to G$. Zeichnen Sie den hierdurch definierten Baum.

7. Gegeben sei folgender Pfad: $A \to B \to C \to A$. Gehört dieser zu einem Baum? Begründen Sie Ihre Antwort.

Programmieren 1 WS2x/2x Prof. Dr. S. Kruse	Name	
	Semester	
	Matrikel-Nr.	

2. Aufgabe: Kundenstatistik (20 Punkte)

Als Betreiber eines Elektronikfachhandels sind Sie an der Erstellung einer Kundenstatistik interessiert, welche Ihnen Aufschluss darüber geben soll, wie häufig Vertreter unterschiedlicher Altersgruppen Ihren Fachhandel besuchen. Schreiben Sie ein Programm, in dem mit Hilfe der IOLib in einer Schleife das Alter einer Reihe von Besuchern eingelesen wird und die Besucher dabei entsprechend des gelesenen Alters in eine der folgenden Gruppen eingeordnet werden:

Alter	Altersgruppe
10-19	Teenager
20-29	Twens
30-64	Middle Age
>64	Seniors

Altersangaben von 0 bis 9 Jahren werden vom Programm ignoriert. Das Einlesen wird durch Eingabe einer negativen Altersangabe terminiert. Das Programm gibt dann den prozentualen Anteil der jeweiligen Altersgruppe an der Gesamtzahl aller Besucher an sowie deren Gesamtzahl und wird danach beendet.

Beispielablauf:

Eingabeschritte über die IOLib:

```
Your age: 17
Your age: 24
Your age: 32
Your age: 70
Your age: -1
```

Ausgabe am Ende in die Processing-Konsole:

```
Teens: 25.00
Twens: 25.00
Middle Age: 25.00
Seniors: 25.00
Visitors: 4
```

3. Aufgabe: Felder (15 Punkte)

1. Ist das folgende C-Programm korrekt? Markieren und korrigieren Sie eventuelle Fehler.

```
class Feld {

  int data[2];

  void initialisiereFeld (int data) {
    data[1] = 7;
    data[2] = 13;
    data[3] = 22;
  }
}
```

2. Schreiben Sie ein Programm, welches ein gegebenes eindimensionales Feld `a1` mit gerader Anzahl von Feldelementen mit Hilfe einer einzelnen Schleife so in ein neues eindimensionales Feld `a2` überführt, dass in der linken Hälfte des neuen Felds, von links nach rechts, die Summe der gegenüberliegenden Feldelemente aus `a1` steht und in der rechten Hälfte von `a2`, von rechts nach links, deren Differenz. Dabei liegt in `a1` das erste Feldelement dem letzten gegenüber, das zweite dem vorletzten, usw.

   Für das Beispielfeld `a1`, bestehend aus den Elementen `[2][5][7][4][12][-3][8][-2]`, generiert das Programm dann das folgende Feld `a2`: `[0][13][4][16][-8][10][-3][4]`.

   Verwenden Sie in Ihrem Programm das Beispielfeld `a1`.

Programmieren 1 WS2x/2x Prof. Dr. S. Kruse	Name	
	Semester	
	Matrikel-Nr.	

4. Aufgabe: Klassen (35 Punkte)

Im großen Dünenmeer befinden sich ein Kameltreiber und ein Kamel mit unterschiedlichen Waren auf Karawanen zwischen den Oasen. Für jede Karawane benötigt das Kamel 80l Wasser. Erreicht es eine Oase und beträgt sein Wasservorrat weniger als 80l, so trinkt es, bis es wieder 200l Wasser aufgenommen hat. Dann geht es auf die nächste Karawane und so geht es weiter bis ans Ende der Zeit.

1. Definieren Sie eine Klasse `Kameltreiber`. Kameltreiber besitzen einen Namen, der bei ihrer Erzeugung angegeben wird. Daneben können sie mit Kamelen auf Karawanen gehen:

   `public void karawane(Kamel k, String ware)`

   Zu Beginn einer Karawane wird das Kamel mit Ware beladen. Um welche Ware es sich handelt, wird mit der Zeichenkette `ware` festgelegt und in die Processing-Konsole ausgegeben. Siehe hierzu die Beispielausgaben unten.

2. Definieren Sie eine Klasse `Kamel`. Diese verfügen über folgende Fähigkeiten:

   - Kamele können `public void beladen(String ware)` werden. Sie merken sich die Ware intern.
   - Kamele können `private void trinken()`. Durch das Trinken erhöht ein Kamel seinen Wasservorrat auf 200l. Nach seiner Erzeugung verfügt ein Kamel schon über 200l Wasser.
   - Kamele können auf eine `public void karawane()` gehen. Sie geben dann die transportierte Ware und ihren Wasservorrat nach Abschluss der Karawane in die Processing-Konsole aus. Jede durchwanderte Karawane verringert ihren Wasservorrat um 80l. Ist der Wasservorrat am Ende einer Karawane kleiner als 80l, so trinkt das Kamel.

3. Schreiben Sie ein Hauptprogramm, in dem ein Kameltreiber mit dem Namen `Camillo` sowie ein Kamel erzeugt werden. Danach gehen die beiden in einer Endlosschleife auf Karawane, wobei abwechselnd Datteln und Feigen transportiert werden.

Simulieren Sie die in den Methoden auftretenden Handlungen mittels Textausgaben in das Ausgabefenster, entsprechend zu der Programmausgabe unten. Nutzen Sie bei den Textausgaben soweit wie möglich in dem jeweiligen Objekt gespeicherte Informationen. Halten Sie sich streng an die oben definierten Methodenköpfe.

Programmausgabe:

```
Camillo: Belade Kamel mit Datteln
Kamel: Karawane mit Datteln. 120l nach Ankunft
Camillo: Belade Kamel mit Feigen
Kamel: Karawane mit Feigen. 40l nach Ankunft
Kamel: Trinke
Camillo: Belade Kamel mit Datteln
Kamel: Karawane mit Datteln. 120l nach Ankunft
Camillo: Belade Kamel mit Feigen
Kamel: Karawane mit Feigen. 40l nach Ankunft
Kamel: Trinke
...usw.
```