Universität Augsburg
Institut für Informatik

Übung zur Vorlesung Informatik 1 WiSe 2022/2023
Fakultät für Angewandte Informatik
Institut für Informatik
Dr. M. Frieb, Dr. J. Mische, V. Schmid, L. Petrak, P. Schalk 11.11.2022

Übungsblatt 4

Abgabe: 21.11.2022 09:00 Uhr

- Dieses Übungsblatt muss im Team abgegeben werden (Einzelabgaben sind nicht erlaubt!).
- Die **Zeitangaben** geben zur Orientierung an, wie viel Zeit für eine Aufgabe später in der Klausur vorgesehen wäre; gehen Sie davon aus, dass Sie zum jetzigen Zeitpunkt wesentlich länger brauchen und die angegebene Zeit erst nach ausreichender Übung erreichen.

* leichte Aufgabe / ** mittelschwere Aufgabe / *** schwere Aufgabe

Offener Zoom-Arbeitsraum (OZR):
Da zwischenzeitlich ein falscher Link zum offenen Zoom-Raum veröffentlicht war, finden Sie hier den korrekten Link:

https://uni-augsburg.zoom.us/j/97733319417?pwd=RnZjOVhpZktINlVKTlVPcm1uTEEyUT09

Der offene Zoom-Raum findet immer freitags von 14:00 bis 17:00 Uhr statt. Dort sind ein Mitarbeiter und einige Tutoren anwesend, denen Sie Fragen zum aktuellen Stoff und zum aktuellen Übungsblatt stellen können. Außerdem können Sie dort in Breakout-Sessions mit Ihrem Team oder anderen Studierenden gemeinsam am Übungsblat arbeiten.

Aufgabe 13 * *(1K- und 2K-Codierung und -Decodierung, 26 Minuten)*

a) *(*, 3 Minuten)* Decodieren Sie die folgenden Bitmuster in 1-Komplement-Codierung mit 8 Bits. Sie dürfen als Hilfestellung für den Anfang die folgende Schablone ausschneiden und benutzen:

-127	64	32	16	8	4	2	1
☐	☐	☐	☐	☐	☐	☐	☐

(i)	0	1	0	0	1	0	1	1
(ii)	1	1	0	0	0	0	1	1
(iii)	1	0	1	0	0	1	0	1

b) *(*, 3 Minuten)* Decodieren Sie die folgenden Bitmuster in 2-Komplement-Codierung mit 8 Bits. Sie dürfen als Hilfestellung für den Anfang die folgende Schablone ausschneiden und benutzen:

-128	64	32	16	8	4	2	1
☐	☐	☐	☐	☐	☐	☐	☐

(i)	0	0	1	0	1	0	1	0
(ii)	0	1	1	0	0	0	1	1
(iii)	1	1	1	1	0	0	1	0

c) *(*, 20 Minuten)* Befüllen Sie die leeren Zellen in folgender Tabelle. Sollte x außerhalb des Definitionsbereichs einer Codierung liegen, so schreiben Sie ein Kreuz $\times$ in die Zelle.

Dezimalzahl x	$c_{1K,4}(x)$	$c_{2K,4}(x)$	$c_{1K,8}(x)$	$c_{2K,8}(x)$
7				
	1100			
		1010		
			0000 1000	
				0010 0001
-128				
				1001 1001

Sie müssen Ihre Ergebnisse nicht begründen.

Aufgabe 14 ** *(Rechnen mit 1K- und 2K-Codierungen, 24 Minuten)*

Führen Sie folgende Rechenoperationen auf den gegebenen Bitmustern aus:

Rechnung	**Ergebnis**	**Ergebnis als Dezimal-Zahl**
$1011 \oplus_{1K,4} 0001 =$		
$1001 \ominus_{1K,4} 0010 =$		
$0101 \ominus_{1K,4} 1101 =$		
$0011 \oplus_{1K,4} 0101 =$		
$1001 \oplus_{2K,4} 0110 =$		
$0110 \oplus_{2K,4} 0111 =$		
$1001 \ominus_{2K,4} 0010 =$		
$0101 \ominus_{2K,4} 1110 =$		
$0100\,1011 \ominus_{1K,8} 0110\,1100 =$		
$1001\,1001 \oplus_{1K,8} 0111\,0011 =$		
$1101\,0111 \oplus_{2K,8} 1000\,0100 =$		
$0001\,1011 \ominus_{2K,8} 0110\,0001 =$		

Geben Sie jeweils die zugehörige Rechnung an, sowie bei jeder Rechnung, ob ein *Bereichsüberlauf im positiven Bereich*, ein *Bereichsüberlauf im negativen Bereich* oder *keines von beidem* stattgefunden hat.

Aufgabe 15 ** *(Kleine Induktions-Beweise, 20 Minuten)*

Beweisen Sie die folgenden Aussagen mittels vollständiger Induktion:

a) *(**, 10 Minuten)* Für jedes $n \in \mathbb{N}$ gilt:

$$n < 2^n$$

b) *(***, 10 Minuten)* Für jedes $n \in \mathbb{N}$ gilt:

$$\sum_{i=1}^{n} \frac{1}{i \cdot (i+1)} = \frac{n}{n+1}$$

Aufgabe 16 ** *(Automatisierung für 1K- und 2K-Darstellung, 20 Minuten)*

a) *(**, 10 Minuten)* Schreiben Sie eine Funktion mit dem Prototyp
`void zwei_k_add(int[] a, int[] b, int[] result, int len)`,
die

- erwartet, dass in `a` und `b` jeweils das Ergebnis der Codierung einer ganzen Zahl nach der 2K-Codierung in `len` Bits beginnend mit dem vordersten Bit gespeichert ist. z.B. ist die Zahl $c_{2K,4}(-8) = 1000$ als `a[0] = 1`, `a[1] = 0` usw. gespeichert.
- das Ergebnis der Addition in 2K-Codierung in `result` speichert.

Testen Sie Ihre Funktion in der Haupt-Funktion mit passenden Feldern.

b) *(**, 5 Minuten)* Wie müsste man die Funktion aus Teilaufgabe a) anpassen, damit das Ergebnis der Addition in 1K-Codierung berechnet wird?

c) *(**, 5 Minuten)* Schreiben Sie ein ausführbares C-Programm, das einer `char`-Variable c einen zufälligen, gültigen Wert zuweist und dann mit eine `for`-Schleife für jedes Bit von c *ausschließlich* unter Verwendung bitweiser Operatoren ausgibt, ob es gesetzt ist oder nicht.