

KLAUSUR im FACH *Ingenieur-Informatik* im Wintersemester 2018/2019

Name, Vorname:

Studiengang/Semester:

Klausureinsicht:

Datum, Uhrzeit: Unterschrift:

Prüfer:

Dipl.-Ing. (FH) Sebastian Balz, Dipl.-Ing. (FH) Marko Weber,

Dipl.-Ing. (FH) Rüdiger Ehret, Prof. Dr.-Ing. Daniel Fischer

Bearbeitungshinweise

1. Tragen Sie auf jeder Seite in der Kopfzeile Ihre Matrikelnummer ein.
2. Der Aufgabensatz (inkl. Deckblatt und Anhang), der aus 12 Seiten besteht (Seite 1 bis 12), ist auf Vollständigkeit zu überprüfen.
3. Der Aufgabensatz ist mit den Lösungsblättern abzugeben.
4. Lösungen auf selber mitgebrachten Lösungsblättern werden nicht ausgewertet. Verwenden Sie die Ihnen ausgeteilten Lösungsblätter und tragen Sie auch dort Ihre Matrikelnummer ein.
5. Bei Rechenaufgaben muss der Lösungsweg ersichtlich und lesbar sein, sonst erfolgt keine Bewertung der Aufgabe oder des Aufgabenteils.
6. Die Bearbeitungszeit beträgt 90 Minuten.
7. Es wird hiermit darauf hingewiesen, dass vom Prüfungsamt nicht vorher geprüft wurde, ob Sie das Recht bzw. die Pflicht zur Teilnahme an dieser Klausur haben. Die Teilnahme erfolgt auf eigene Gefahr, gleichzeitig bekundet die Teilnahme die Zustimmung zu diesem Passus.
8. Hilfsmittel:
 - Handgeschriebene Formelsammlung (1 DIN A4 Blatt)
 - C-Coding Styleguide ohne Anmerkungen – Markierungen erlaubt (selbst mitbringen)
 - Taschenrechner sind **nicht** erlaubt

9. Die Nichteinhaltung des C-Coding Styleguides führt zu Punktabzug**10. Bewertung:**

Gesamtpunktzahl: 100 Punkte (10% Überhang)

Note 1,0: 90 Punkte

Note 4,0: 45 Punkte

Aufgabe	1	2	3	4	5	6	7	SUMME	
Punkte	10	20	10	10	10	10	30	100	NOTE
Erreichte Punkte									

Aufgabe 1: (10 Punkte)

Es ist jeweils **nur** eine Antwort pro Frage richtig. Falsche Antworten führen **nicht** zu Punktabzug! Kreuzen Sie auf alle Fälle immer eine Antwort an!

1.1 Was wird als kleinste Informationseinheit bezeichnet?

- a) char
- b) Byte
- c) Algorithmus
- d) Pointer
- e) Bit
- f) Keine Antwort ist richtig

1.2 Welche Funktion ist in der Utilities-Bibliothek (Utilities.lib, die im Labor Ingenieur-Informatik verwendet wurde) enthalten?

- a) printf
- b) _getch
- c) scanf
- d) scanf_s
- e) strcpy
- f) Keine Antwort ist richtig

1.3 Was ist **kein** C-Schlüsselwort?

- a) goto
- b) decimal
- c) continue
- d) for
- e) static
- f) Keine Antwort ist richtig

1.4 Der Wertebereich einer integer Variablen

`int iA = 42;`

ist auf der im Labor verwendeten Plattform (32-Bit-System):

- a) -128 bis 127
- b) -128 bis 128
- c) -32768 bis 32767
- d) -2147483648 bis 2147483647
- e) 0 bis 4294967295
- f) Keine Antwort ist richtig

1.5 Die Schlüsselworte signed, unsigned, short und long werden als ... bezeichnet.

- a) Modifizierer
- b) Operatoren
- c) Basisdatentypen
- d) Literale
- e) Casts
- f) Keine Antwort ist richtig

1.6 Die folgende Zeile findet sich am Anfang in einer c-Datei (außerhalb einer Funktion, 32-Bit-System):

```
static int iB = 0xBADDCAFE;
```

Welche Aussage ist richtig?

- a) Es handelt sich um eine Deklaration einer programmglobalen Variablen
- b) Es handelt sich um eine Definition einer programmglobalen Variablen
- c) Es handelt sich um eine Deklaration einer modulglobalen Variablen
- d) Es handelt sich um eine Definition einer modulglobalen Variablen
- e) Die Compiler erzeugt einen Fehler, da man einer int-Variablen keine Zeichenkette zuweisen kann.
- f) Keine Antwort ist richtig

1.7 Wie lässt sich die Fließkommavariablen `double dVal` auf der Konsole ausgeben?

- a) `printf("%d", &dVal);`
- b) `printf("%d", dVal);`
- c) `printf("%f", &dVal);`
- d) `printf("%f", *dVal);`
- e) `printf("%dVal", dVal);`
- f) Keine Antwort ist richtig

1.8 Wie lautet die Definition eines Funktionszeigers, der auf die Funktion `strcpy` zeigen soll?

- a) `int (*moo) (int, char*);`
- b) `char* (*)goo (char[], char);`
- c) `char* (*foo) (char*, const char*);`
- d) `void* (*hoo) (char* pac1[], ...);`
- e) `void* functionpointertostrcpy(void);`
- f) Keine Antwort ist richtig

1.9 Welchen Wert hat `c%` der Funktion *TowersOfHanoi* aus Anhang B?

- a) 1
- b) 2
- c) 3
- d) 4
- e) 8
- f) Keine Antwort ist richtig

1.10 Welchen Wert hat `v(G)` der Funktion *TowersOfHanoi* aus Anhang B?

- a) 1
- b) 2
- c) 3
- d) 4
- e) 8
- f) Keine Antwort ist richtig

Aufgabe 2: (20 Punkte)

2.1 Führen Sie eine Multiplikation ($1010 \cdot 1001$) im Binärsystem (Dualsystem) anhand des folgenden Schemas aus dem Lernmodul „Zahlensysteme“ durch: (5 Punkte)

0 0 0 0 1 0 1 0	0 0 0 0 1 0 0 1	
		+
Ergebnis:		

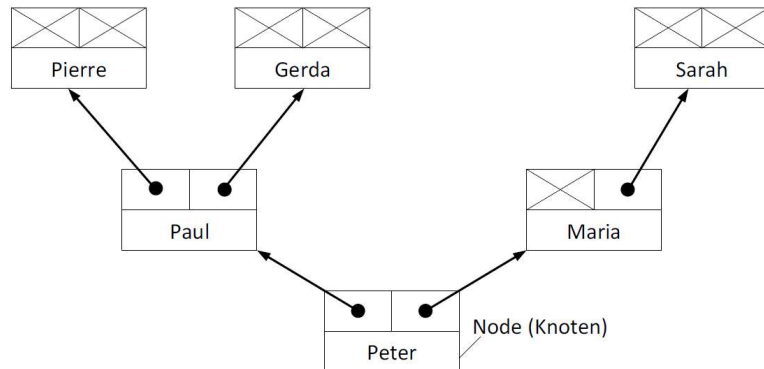
2.2 Zeichnen Sie das Strukturdiagramm (Structure Chart) der Anwendung aus dem Anhang B. Bibliotheksfunktionen sind nicht zu berücksichtigen. (5 Punkte)

Hinweis: 2 Punkte Abzug für jeden Fehler und jedes fehlende Element. Negative Punkte sind für diese Aufgabe nicht möglich.

2.3 In einem C-Programm soll der Stammbaum einer Person wie folgt abgebildet werden. Deklarieren Sie hierzu eine komplexe Datenstruktur (max. Länge des Namens: 20), mit welcher man einen solchen Stammbaum aufbauen kann! (3 Punkte)

Komplexe Datenstruktur

Stammbaum



2.4 Allokieren Sie **dynamisch** ein Array von drei Variablen (Vater sei Paul, Mutter sei Maria, Kind sei Peter) der obigen komplexen Datenstruktur, initialisieren Sie die Variablen und verbinden Sie die drei Variablen miteinander, damit ein kleiner Stammbaum mit drei Personen entsteht. Geben Sie den Speicher anschließend wieder frei! (7 Punkte)

Aufgabe 3: (10 Punkte)

Schreiben Sie hinter die Kommentare, was *exakt* in der Konsole auf einem 32-Bit System ausgegeben wird. Im Kommentar sind die möglichen Punkte aufgeführt.
Hinweis: Hexadezimale Ziffer A-F werden bei printf als Kleinbuchstaben ausgegeben.

```
#include "MyVars.h"
```

```
#define SETBIT(x,n) ((x)|=1<<(n))
#define EVEN(x) ((x)&1)

enum StudyProgram {MK, MKplus, EI3nat};

struct Flags
{
    unsigned int uiVal1 : 17;
    unsigned int uiVal2 : 16;
};
typedef struct Flags sFlags_t;
typedef sFlags_t* psFlags_t;

union Trans
{
    struct Flags* psMyFlags;
    double dVal;
};
```

MyVars.h

```
void AG3(void)
{
    unsigned char uc1;
    unsigned char uc2;
    enum StudyProgramm eMyProgram = MK;
    char acName[] = "HS Offenburg";
    char* pc1;

    printf("%d\n", (19%8)-(20/8));           // _____ (1P)
    printf("%d\n", sizeof(eMyProgram));      // _____ (1P)
    printf("%d\n", sizeof(struct Flags));    // _____ (1P)
    printf("%d\n", sizeof(psFlags_t));       // _____ (1P)
    printf("%d\n", sizeof(union Trans));     // _____ (1P)
    pc1 = strtok(acName, "Offenburg");
    printf("%d\n", strlen(pc1));             // _____ (1P)

    uc1 = 0xA5;
    uc2 = 0xFF;
    printf("%x\n", ((~uc1)&uc2));             // _____ (1P)
    printf("%x\n", EVEN(uc1));               // _____ (1P)
    printf("%x\n", SETBIT(uc1,1));           // _____ (2P)

}
```

Aufgabe 4: (10 Punkte)

Im folgenden Programmcode (Dateien bubblesort.h und bubblesort.c) sind **fünf** Fehler enthalten. Korrigieren Sie diese Fehler.

bubblesort.h

```
void BubbleSort(char* pc);
```

bubblesort.c

```
#include <string.h>
#include "bubblesort.h"

static void DoSwap();

void BubbleSort(char* pc)
{
    unsigned int uiLengthStr;
    unsigned int uiX;
    unsigned int uiY;

    if (pc != NULL);
    {
        uiLengthStr = sizeof(pc);

        for (uiX = 0; uiX < (uiLengthStr-1U); uiX++)
        {
            for (uiY = 0; uiY < (uiLengthStr-uiX-1U); uiY++)
            {
                if ((pc+uiY) > (pc+uiY+1U))
                {
                    DoSwap((pc+uiY), (pc+uiY+1U));
                }
            }
        }
    }
}

static void DoSwap(char* pc1, char* pc2)
{
    char cDum;

    cDum = *pc2;
    *pc1 = *pc2;
    *pc2 = cDum;
}
```

Aufgabe 5: (10 Punkte)

Schreiben Sie C-Code (keine Funktionen, nur reiner C-Code, Variablendefinition nicht vergessen), welcher eine vorzeichenlose Integerzahl `uiN` einliest. `uiN` soll kleiner als 24 sein, ansonsten ist eine Fehlermeldung auszugeben. Für beispielsweise `uiN = 5` soll folgende Figur rechts mittels dreier Schleifen und **ohne** weitere `if`-Abfrage auf der Konsole ausgegeben werden.

YYYYX
YYYXX
YYXXX
YXXXX
XXXXX

Aufgabe 6: (10 Punkte)

Gegeben sei die folgende Summenformal

$$\sum_{k=0}^n \frac{(-1)^k}{k+1} = \frac{1}{1} - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} - \dots \quad k, n \in N_0$$

Implementieren Sie eine rekursive Funktion `CalcSumEquation`, welche die Summe der ersten n Elemente der obigen Reihe **rekursiv** berechnet. k hat dabei die Werte 0 bis n .

Beispiele für $n=0$, $n=1$ und $n=2$ sind:

$$\sum_{k=0}^{n=0} \frac{(-1)^k}{k+1} = \frac{(-1)^0}{0+1} = \frac{1}{1}$$

$$\sum_{k=0}^{n=1} \frac{(-1)^k}{k+1} = \frac{(-1)^0}{0+1} + \frac{(-1)^1}{1+1} = \frac{1}{1} - \frac{1}{2}$$

$$\sum_{k=0}^{n=2} \frac{(-1)^k}{k+1} = \frac{(-1)^0}{0+1} + \frac{(-1)^1}{1+1} + \frac{(-1)^2}{1+2} = \frac{1}{1} - \frac{1}{2} + \frac{1}{3}$$

usw.

Hinweis: `double pow(double base, double exponent);`

```
_____ CalcSumEquation(_____)
{
```

```
    return _____;
}
```

Aufgabe 7: (30 Punkte)

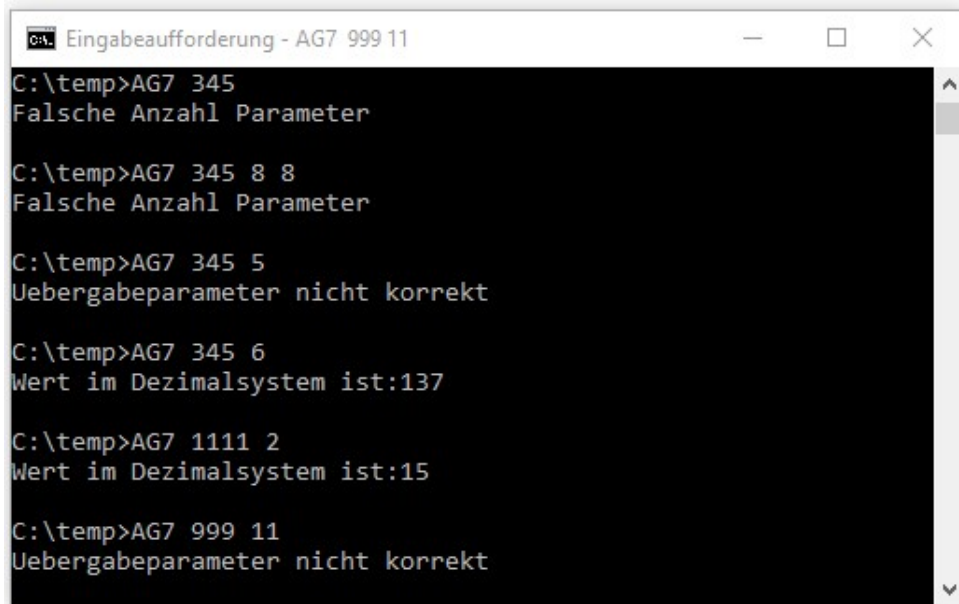
7.1 Implementieren Sie die Funktion `IsStringInRange`, welche bestimmt, ob alle Zeichen der übergebenen Zeichenkette `pc` innerhalb des ASCII-Bereichs `[cMin, cMax]` liegen. Die Funktionsweise von `IsStringInRange` ist wie folgt: (10 Punkte)

```
/**
 * @fn int IsStringInRange(char* pc, char cMin, char cMax)
 * @brief Checks whether pc is valid AND cMax >= cMin
 *        Checks if all characters in pc are in range (ASCII)
 *        cMin and CMax are part of the range
 * @param pc Pointer to char array, 0x00 marks end of string
 * @param cMin Min value
 * @param cMax Max value
 * @return <0 if parameter not okay, 0 if not in Range, 1 if in Range
 */
```

Verwenden Sie zur Lösung den ausgeteilten Prüfungsbogen!

7.2 Schreiben Sie ein vollständiges C-Programm (Name der Exe sei AG7), welches über die Kommandozeile eine vorzeichenlose Zahl in einem Zahlensystem zur Basis `base` sowie `base` übergeben bekommt, wobei `base` größer als 1 und kleiner als 11 sein muss. Überprüfen, ob die Parameter korrekt übergeben wurden. Verwenden Sie hierzu auch `IsStringInRange` aus 7.1. Falls alle Parameter korrekt übergeben wurden, geben Sie die Zahl als Dezimalzahl auf der Konsole aus. (20 Punkte)

Die prinzipielle Funktionsweise von AG7 zeigt die folgende Abbildung.



```
Eingabeaufforderung - AG7 999 11
C:\temp>AG7 345
Falsche Anzahl Parameter

C:\temp>AG7 345 8 8
Falsche Anzahl Parameter

C:\temp>AG7 345 5
Uebergabeparameter nicht korrekt

C:\temp>AG7 345 6
Wert im Dezimalsystem ist:137

C:\temp>AG7 1111 2
Wert im Dezimalsystem ist:15

C:\temp>AG7 999 11
Uebergabeparameter nicht korrekt
```

Hinweise:

- Verwenden Sie `double pow(double dbase, double dexponent)`
- Mögliche Rundungsfehler durch `pow` sind zu vernachlässigen.
- Eine Zahl aus dem Fünfersystem enthält **keine** Ziffer 5

Verwenden Sie zur Lösung den ausgeteilten Prüfungsbogen!

Viel Erfolg!

Anhang A: ASCII-Tabelle

Dez.	Hex.	Zeichen	Dez.	Hex.	Zeichen	Dez.	Hex.	Zeichen	Dez.	Hex.	Zeichen
0	0	NUL	32	20		64	40	@	96	60	`
1	1	SOH	33	21	!	65	41	A	97	61	a
2	2	STX	34	22	"	66	42	B	98	62	b
3	3	ETX	35	23	#	67	43	C	99	63	c
4	4	EOT	36	24	\$	68	44	D	100	64	d
5	5	ENQ	37	25	%	69	45	E	101	65	e
6	6	ACK	38	26	&	70	46	F	102	66	f
7	7	BEL	39	27	'	71	47	G	103	67	g
8	8	BS	40	28	(72	48	H	104	68	h
9	9	HT	41	29)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	
29	1D	GS	61	3D	=	93	5D]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	DEL

Anhang B:

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

void TowersOfHanoi(unsigned int uiRings, char cSource,
                  char cDest, char cDummy);

int main(void)
{
    unsigned uiRings;
    printf("Bitte geben Sie die Anzahl der Ringe ein:");
    scanf("%u", &uiRings);
    printf("\nAlgorithmus\n");
    TowersOfHanoi(uiRings, 'A', 'B', 'C');
    _getch();
    return 0;
}

void TowersOfHanoi(unsigned int uiRings, char cSource,
                  char cDest, char cDummy)
{
    if (uiRings == 0)
    {
        printf("Nothing to do\n");
    }
    else
    {
        if (uiRings == 1)
        {
            printf("Move from %c to %c\n", cSource, cDest);
        }
        else
        {
            TowersOfHanoi(uiRings - 1, cSource, cDummy, cDest);
            TowersOfHanoi(1, cSource, cDest, cDummy);
            TowersOfHanoi(uiRings - 1, cDummy, cDest, cSource);
        }
    }
}
```