

Lösungshinweise zur Teilprüfung Software- und Internettechnologie Programmierkurs 2 Sommersemester 2005

Aufgabe 1: Verständnisfragen

- a) Ausgabe für `int a[] = 1,2,5,7; printFeld(a, 4);` lautet 2 5 7
mögliche korrigierte Funktion:

```
void printFeld(int*a, int n){
    int i=0;
    while(i<n)
        printf("%i ", a[i++]);
}
```

$$\begin{aligned}
 b) \quad a &= \underbrace{(12&\&15)}_{(1)_{10}} + \underbrace{013}_{(11)_{10}} = 014 = (12)_{10} \\
 b &= \underbrace{12}_{(1100)_2} \& \underbrace{0x11}_{(00010001)_2} = 0 \\
 c &= \underbrace{(a \wedge 013)}_{(1100)_2 \wedge (1011)_2} \mid \underbrace{15}_{(1111)_2} = \underbrace{15}_{(1111)_2} \\
 d &= c \gg \underbrace{(b + 1)}_1 = (0111)_2 = 7
 \end{aligned}$$

- c) `void erhoeheGehalt(Angest* angest, int bonus){`
`angest->gehalt += bonus;`
`}`

Beispielhafter Aufruf: `erhoeheGehalt(&klaus,15)`

- d) `mov 0(r10),r15`
`add #2,r10`

Aufgabe 2: C-Programmierung

```
#include <stdio.h>
#include <stdlib.h>
int main(void){

    int n;                /* Anzahl der Iterationen */
    unsigned long m;     /* Indexvariable */
    unsigned long fak_m; /* m! */
    double e=0;         /* Naecherung fuer e */

    /* Anzahl der Iterationen einlesen */
    printf("Anzahl der Iterationen : ");
    scanf("%u", &n);
    if (n < 1){
        printf("Anzahl der Iterationen muss groesser als Null sein.\n");
        return 1;
    } /* if */
    /* e berechnen */
    for (m=0; m < n ; m++){

        if (m==0){
```

```

    fak_m=1;
} /* if */
else{
    fak_m=fak_m*m;
} /* else */

e = e + 1/(double)fak_m;

} /* for */

/* Ergebnis ausgeben */
printf("Eulersche Zahl e ist etwa %1.8f\n", e);
return 0;

} /* main */

```

Aufgabe 3: Dynamische Datenstrukturen

```

a) knoten* neuerKnoten(unsigned int neuerWert, knoten* links, knoten* rechts){
    knoten *new;
    if ((links!=NULL && links->wert > neuerWert) ||
        (rechts!=NULL && rechts->wert <= neuerWert)) return NULL;
    new = (knoten*) malloc(sizeof(knoten));
    if (new == NULL) return NULL;
    new->wert = neuerWert;
    new->linkerSohn = links;
    new->rechterSohn = rechts;
    return new;
}

b) void listeAbsteigend(knoten* b, void (*print)(unsigned int z)){
    if (b==NULL) return;
    listeAbsteigend(b->rechterSohn, print);
    print(b->wert);
    listeAbsteigend(b->linkerSohn, print);
}

c) int zaehle(knoten* b, unsigned int z){
    if (b==NULL) return 0;
    if (z > b->wert){
        return zaehle(b->rechterSohn, z);
    } /* if */
    return (b->wert == z) + zaehle(b->linkerSohn, z);
}

```

Aufgabe 4: Sensor-knotensteuerung

```

.alert:
    bis.b    #0x7,&0x0029    ; alle LEDs ausschalten
.loop:    bic.b    #0x1,&0x0029    ; rot einschalten
    bis.b    #0x2,&0x0029    ; gruen ausschalten
    mov     #0xffff,r14    ; Wartedauer setzen und warten
    mov     #0,r15
    call    #wait
    bic.b    #0x2,&0x0029    ; gruen einschalten
    bis.b    #0x1,&0x0029    ; rot ausschalten
    mov     #0xffff,r14    ; Wartedauer setzen und warten
    mov     #0,r15
    call    #wait
    jmp     .loop

```

Aufgabe 5: MSP430-Assembler

```
a)  for(i = 0; i < lengthA + lengthB; i++)
    if (aPos == lengthA)
        c[i] = b[bPos++];
        continue;
    if (bPos == lengthB)
        c[i] = a[aPos++];
        continue;
    if (a[aPos] < b[bPos])
        c[i] = a[aPos++];
    else c[i] = b[bPos++];

merge:
    ;; Paramter:
    ;; r8  Startadresse Feld a
    ;; r9  lengthA
    ;; r10 Startadresse Feld b
    ;; r11 lengthB
    ;; r12 Startadresse Ergebnisfeld c

    ;; lokale Variablen:
    ;; r4  i
    ;; r5  endPosA
    ;; r6  endPosB
    ;; r7  lengthA + lengthB

    push    r4          ; Registerinhalte retten
    push    r5
    push    r6
    push    r7

    mov     r8,r5       ; Endposition Feld a initialisieren
    add     r9,r5
    add     r9,r5
    mov     r10,r6      ; Endposition Feld b initialisieren
    add     r11,r6
    add     r11,r6
    mov     r9,r7       ; r7 = lengthA + lengthB
    add     r11,r7

for_i:  mov     #0,r4     ; i=0
        cmp     r7,r4     ; (lengthA + lengthB) - i berechnen
        jeq     endfor    ; if (i==lengthA + lengthB) goto loopend
if_aPos: cmp     r5,r8     ; if (a!=endposA)
        jne     if_bPos   ; goto b_pos
        mov     @r10+,0(r12) ; *c = *(b++)
        jmp     loopend
if_bPos: cmp     r6,r10    ; if (b!=endPosB)
        jne     if_ab     ; goto ifab
        mov     @r8+,0(r12) ; *c = *(a++)
        jmp     loopend
if_ab:  cmp     0(r10),0(r8) ; if (*a >=*b)
        jge     else      ; goto else
        mov     @r8+,0(r12) ; *c = *(a++)
        jmp     loopend
else:   mov     @r10+,0(r12) ; *c = *(b++)
loopend: add     #1,r4     ; i++
        add     #2,r12    ; c++
        jmp     for_i

endfor: pop     r7        ; Registerinhalte wiederherstellen
        pop     r6
        pop     r5
        pop     r4
        ret
```