Name: Matrikelnummer:

Lehrstuhl für Praktische Informatik IV
Prof. Dr. Wolfgang Effelsberg
A5, Raum B223
68131 Mannheim
Telefon: (0621) 181–2600
Email: effelsberg@informatik.uni-mannheim.de

Marcel Busse
A5, Raum B221
68131 Mannheim
Telefon: (0621) 181–2616
Email: busse@informatik.uni-mannheim.de

UNIVERSITÄT MANNHEIM

Programmierkurs in C für Bachelor IMI — Klausur
Herbst-/Wintersemester 2007 — 10. Dezember 2007

Hinweise

1. Überprüfen Sie bitte Ihr Klausurexemplar auf Vollständigkeit (7 Seiten).
2. Unterschreiben Sie die Klausur auf der Rückseite des letzten Blatts.
3. Bearbeiten Sie die Aufgaben *ausschließlich* auf dem Aufgabenblatt der jeweiligen Aufgabe. Benutzen Sie ggf. auch die Rückseite der Aufgabenblätter.
4. Schreiben Sie auf jedes Blatt, das bewertet werden soll, oben Ihren Namen und Ihre Matrikelnummer.
5. Verwenden Sie nur dokumentenechte Stifte (z. B. keinen Bleistift) und keine roten Stifte.
6. Es sind keine Hilfsmittel zugelassen.
7. Die Bearbeitungszeit beträgt 33 Minuten.

Korrekturzeile — Bitte *nicht* ausfüllen!

Aufgabe	1	2	3	Summe
Max. Punktzahl	9	9	15	33
Erreichte Punktzahl				

Name: Matrikelnummer:

Aufgabe 1 — 9 Punkte

Aufgabe 1 a) — 3 Punkte

Implementieren Sie eine Funktion **void** swap(**int**∗ a, **int**∗ b), die die Inhalte der beiden Zeiger a und b vertauscht.

Lösung:

```
void swap(int* a, int* b) {
  int tmp;
  tmp = *a;
  *a = *b;
  *b = tmp;
}
```

Aufgabe 1 b) — 6 Punkte

Gegeben sei die Funktion

```
int unknown(int* a, int n) {
  int i;
  int* p;
  p = a;
  for (i = 0; i < n; i++) {
    if (a[i] > 5) {
      p++;
    }
  }
  return (p-a);
}
```

Wie lautet der Rückgabewert der Funktion mit dem Aufruf unknown(a, 10), wenn sie auf das Feld a angewendet wird, das wie folgt initialisiert ist:

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
5	4	6	7	6	3	8	2	1	10

Lösung:

Name: Matrikelnummer:

5

Aufgabe 2 — 9 Punkte

Aufgabe 2 a) — 4 Punkte

Implementieren Sie eine Funktion, die das innere Produkt zweier Vektoren berechnet und zurückliefert. Das innere Produkt zweier Vektoren a, b mit den Komponenten

$$a_1, a_2, a_3, \ldots, a_n \quad \text{und} \quad b_1, b_2, b_3, \ldots, b_n$$

ist definiert als

$$a \cdot b \;=\; a_1 \cdot b_1 + a_2 \cdot b_2 + a_3 \cdot b_3 + \ldots + a_n \cdot b_n.$$

Lösung:

```
int product(int* a, int* b, int n) {
  int i;
  int sum;
  sum = 0;
  for (i = 0; i < n; i++) {
    sum += a[i]*b[i];
  }
  return sum;
}
```

Aufgabe 2 b) — 5 Punkte

Implementieren Sie eine Funktion

void diagram(**int*** v, **int** n),

die die Werte des Vektors v der Länge n als Balkendiagramm entsprechend dem Beispiel

```
0 ( 0)|
1 ( 1)|*
2 ( 3)|***
3 ( 5)|*****
4 ( 7)|*******
5 (10)|**********
```

Name: Matrikelnummer:

```
 6 ( 9)|*********
 7 ( 6)|******
 8 ( 4)|****
 9 ( 2)|**
10 ( 0)|
```

ausgibt.

Hinweis: Die Werte müssen vorher nicht sortiert werden.

Lösung:

```
void diagram(int* v, int n) {
  int i, j;
  for (i = 0; i < n; i++) {
    printf("%2d (%2d)|", i, v[i]);
    for (j = 0; j < v[i]; j++) {
      printf("*");
    }
    printf("\n");
  }
}
```

Aufgabe 3 — 15 Punkte

Im Folgenden soll eine aufsteigend sortierte und doppelt verkettete Liste implementiert werden. Hierfür verwenden wir folgende Datenstruktur:

```
typedef struct node {
  int data;            /* Inhalt des Knotens */
  struct node* prev;   /* Zeiger auf den Vorgaengerknoten */
  struct node* next;   /* Zeiger auf den Nachfolgerknoten */
} list_t ;

list_t * list ;
```

Aufgabe 3 a) — 10 Punkte

Implementieren Sie eine Funktion

```
list_t * insert( list_t * list , int value),
```

die einen int-Wert in die übergebene Liste (sortiert) einfügt und das erste Element der Liste zurückliefert.

Name: Matrikelnummer:

Lösung:

```
list_t * insert( list_t * list , int value) {
  list_t * tmp;
  list_t * p = list;
  if ( list == NULL) {
    list = ( list_t *)malloc(sizeof( list_t ));
    if ( list == NULL) return NULL;
    list ->data = value;
    list ->prev = list->next = NULL;

  } else {
    while ((p->next != NULL) && (p->next->data < value)) {
      p = p->next;
    }
    tmp = (list_t*)malloc(sizeof( list_t ));
    if (tmp == NULL) return NULL;
    tmp->data = value;
    tmp->prev = p;

    if (p->next == NULL) {
      tmp->next = NULL;
      p->next = tmp;

    } else {
      p->next->prev = tmp;
      tmp->next = p->next;
      p->next = tmp;
    }
  }
  return list
}
```

Aufgabe 3 b) 5 Punkte

Implementieren Sie eine Funktion

list_t * merge(list_t * list_1 , list_t * list_2),

die aus zwei sortierten Listen eine neue (sortierte) Liste erstellt und diese zurückliefert.

*Hinweis: Verwenden Sie hierfür die Funktion list_t * insert(list_t * list , **int** value) aus der vorherigen Aufgabe.*

Name: Matrikelnummer:

Lösung:

```
list_t * merge( list_t* list_1 , list_t * list_2 ) {
  list_t * p;
  list_t * list_3 = NULL;
  p = list_1 ;
  while (p->next != NULL) {
    list_3 = insert( list_3 , p->data);
    p = p->next;
  }
  p = list_2 ;
  while (p->next != NULL) {
    list_3 = insert( list_3 , p->data);
    p = p->next;
  }
  return list_3;
}
```