Nachname: ..... Vorname: ..... MatrNr.: .....

Klausur PR2

HAW-Hamburg, Fakultät Technik und Informatik, Department Informations- und Elektrotechnik
Dr. Robert Heß, 26.1.2009
Bearbeitungsdauer: 90 min
Hilfsmittel: Vorlesungsunterlagen und C/C++ Einführungsbücher (z.B. Kernighan/Ritchie)
Jegliche Art von elektronischen Hilfsmitteln sind nicht erlaubt.

1. Aufgabe (6 Punkte)

Betrachten Sie folgendes Programm:

```
#include <stdio.h>

int main()
{
    char Text[] = "Sehr geehrte Damen und Herren!";
    char *tmp=Text;

    while(*tmp) tmp++;
    while(tmp!=Text) printf("%c", *--tmp);
    printf("\n");
}
```

Begründen Sie, warum in den eckigen Klammern der Variable `Text` keine Zahl stehen braucht:

Was gibt das Programm auf dem Bildschirm aus?

2. Aufgabe (6 Punkte)

Erstellen Sie einen neuen Datentyp mit Namen *tRichtung*, der nur die Zustände *Nord*, *Ost*, *Süd* und *West* einnehmen kann. Hinter den vier Himmelsrichtungen sollen sich die Werte eins bis vier verbergen.

Definieren Sie eine Variable des eben erstellten Typs mit Namen *sunrise* und initialisieren Sie diese mit dem Wert *Ost*.

3. Aufgabe (10 Punkte)

Schreiben Sie ein kleines Programm, welches die Text-Datei *Brief.txt* mit folgendem Inhalt erstellt:

Guten Tag,
bla bla bla.
Auf Wiedersehen!

```
#include <stdio.h>

int main()
{









}
```

4. Aufgabe (9 Punkte)

Eine verkettete Liste hat für seine Kettenelemente folgende Struktur:

```
typedef struct sKette {
    double Wert;
    struct sKette *next;
} tKette;
```

Es soll eine Funktion erstellt werden, die das erste Kettenglied entfernt. Der Funktion wird hierfür die Adresse des Zeigers auf das erste Kettenglied übergeben. Das Element `next` des letzten Kettenglieds hat den Wert `NULL`. Der Speicher wurde dynamisch mit der Funktion `malloc()` reserviert. Bei Erfolg soll der Wert null, bei einem Fehler der Wert –1 zurückgegeben werden.

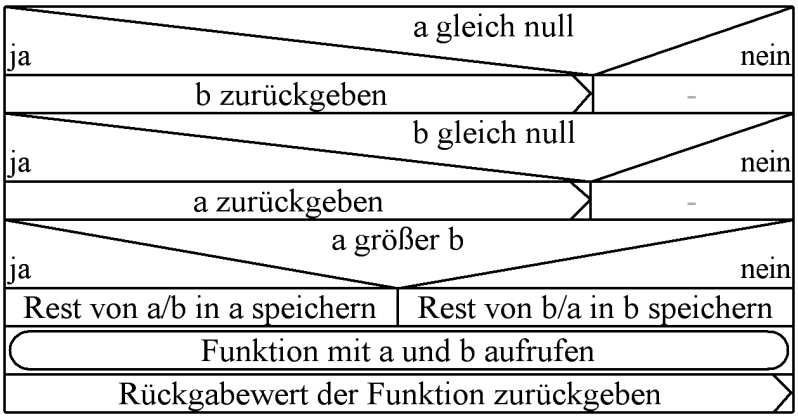
```
int drop(tKette **Start)
{









}
```

5. Aufgabe (14 Punkte)

Gegeben sei folgende Funktionsdeklaration:

```
int Funktion(int a, int b);
```

Die Implementation der Funktion wird durch folgendes Struktogramm beschrieben:



Erstellen Sie die Implementation der Funktion:

```
int Funktion(int a, int b)
{




}
```

Was gibt die Funktion bei folgenden Parametern zurück?

Parameter a	Parameter b	Rückgabewert
12	8	
6	9	
7	8	

Geben Sie der Funktion einen sinnvollen Namen:

.....

6. Aufgabe (10 Punkte)

In das Programm auf der folgenden Seite haben sich in 12 Zeilen Fehler eingeschlichen. Der erste Fehler wurde beispielhaft unten eingetragen. Finden Sie 10 der verbleibenden 11 fehlerhaften Zeilen, und tragen Sie diese mit der dazugehörigen Korrektur und die folgende Tabelle ein.

Zeile	Fehler	Korrektur	
4	*Kommentarende fehlt*	`... 2008 */`	

```
/*  1 */ /* Name der Datei: main.c */
/*  2 */ /* Autor: Robert Heß / E2a */
/*  3 */ /* Version: 1.0 */
/*  4 */ /* Datum: 22.9.2008
/*  5 */
/*  6 */ #jnclude <stdio.h>
/*  7 */
/*  8 */ int Notenanzahl(int Note);
/*  9 */
/* 10 */ int main()
/* 11 */ {
/* 12 */     int Note(6);            /* Anzahl der Noten 1 bis 6 */
/* 13 */     int i;                  /* lokale Laufvariable */
/* 14 */     int bestanden;          /* Anzahl der bestandenen Schüler */
/* 15 */     int nichtBestanden      /* Anzahl der durchgefallenen Schüler */
/* 16 */     double Durchschnitt=0;  /* Notendurchschnitt */
/* 17 */
/* 18 */     /* Überschrift ausgeben */
/* 19 */     printf("   >>> NOTENSPIEGEL <<<\n");
/* 20 */
/* 21 */     /* Anzahl der Noten abfragen */
/* 22 */     for(i=0; i<=6; i++) Note[i] = Notenanzahl(i+1);
/* 23 */
/* 24 */     /* Berechnungen zum Notenspiegel durchführen */
/* 25 */     bestanden = Note[0]+Note[1]+Note[2]+Note[3];
/* 26 */     nichtBestanden = Note[5]+Note[6];
/* 27 */     for(i=0; i<6; i++) Durchschnitt += Note[i]*(i+1);
/* 28 */     if(bestanden+nichtBestanden>0)
/* 29 */         Durchschnitt /= (bestanden+nichtBestanden);
/* 30 */
/* 31 */     /* Ergebnisse ausgeben */
/* 32 */     printf("bestanden:       %d\n", bestanden);
/* 33 */     printf("nicht bestanden: %d\n", nichtBestanden);
/* 34 */     if(Durchschnitt>0)
/* 35 */         printf("Durchschnitt:    %d\n", Durchschnitt);
/* 36 */     else printf("Durchschnitt:    ?\n");
/* 37 */
/* 38 */     /* Schlussmeldung */
/* 39 */     printf("Das war's!\n);
/* 40 */
/* 41 */     return 0;
/* 42 */ }
/* 43 */
/* 44 */ int Notenanzahl(int Note);
/* 45 */ {
/* 46 */     int Anzahl=0;   /* Anzahl der Noten */
/* 47 */     int korrekt=0;  /* zeigt eine korrekte Eingabe an */
/* 48 */
/* 49 */     do
/* 50 */         /* Benutzerabfrage */
/* 51 */         printf("Bitte geben Sie die Anzahl der Note %d ein: ", Note);
/* 52 */         fflush(stdin);
/* 53 */
/* 54 */         /* Test, ob Zahl eingegeben wurde */
/* 55 */         if(scanf("%d", Anzahl)!=1) {
/* 56 */             printf("   Das war keine Zahl!\n");
/* 57 */
/* 58 */         /* Wertebereich testen */
/* 59 */         } else if(Anzahl<0 || Anzahl>999) {
/* 60 */             printf("    Bitte nur der Wertebereich 0 bis 999!\n");
/* 61 */
/* 62 */         /* Eingabe korrekt */
/* 63 */         } else korrekt = 1;
/* 64 */     } while(korrekt);
/* 65 */
/* 66 */     return Anzahl;
/* 67 */ }
```

7. Aufgabe (6 Punkte)

Mit welchen vier Funktionen erfolgt in C die dynamische Speicherverwaltung?

.....

Mit welchen Funktionen/Operatoren erfolgt in C++ die dynamische Speicherverwaltung?

.....

Was wird mit *memory leakage* bezeichnet?

.....

8. Aufgabe (7 Punkte)

Wann wird in C++ der Konstruktor einer Klasse aufgerufen?

.....

Wann wird in C++ der Destruktor einer Klasse aufgerufen?

.....

In einer Klasse gibt es die drei Zugriffsebenen `private`, `protected` und `public`. Auf welche der Ebenen kann innerhalb und außerhalb einer Klasse zugegriffen werden?

Zugriffsbereich	**innerhalb der Klasse**	**außerhalb der Klasse**
`private`	zugänglich nicht zugänglich	zugänglich nicht zugänglich
`protected`	zugänglich nicht zugänglich	zugänglich nicht zugänglich
`public`	zugänglich nicht zugänglich	zugänglich nicht zugänglich