Nachname: ..... Vorname: ..... MatrNo.: .....

Klausur: Programmieren 2

Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Technik und Informatik, Department Informations- und Elektrotechnik
Prof. Dr. Robert Heß, 27.1.2016, Dauer: 180 Min.
Erlaubte Hilfsmittel: Vorlesungsunterlagen, Lösungen aus dem Praktikum und C/C++ Einführungsbücher.

Ergebnis: ..... von 100 Punkten Note: ..... Punkte.

---

1 Einführung

1.1 Worum geht es?

Quiz-Sendungen im Fernsehen, bei denen Kandidaten eine Reihe von Fragen beantworten müssen, sind seit vielen Jahren in Mode und erfreuen sich nach wie vor großer Beliebtheit. In dieser Programmieraufgabe soll ein Programm erstellt werden, dass dem Benutzer in ähnlicher Form Fragen stellt und die Ergebnisse auswertet. Die Fragen werden dem Programm zusammen mit den korrekten Antworten in einer binären Datei zur Verfügung gestellt.

Das Programm liest alle Fragen aus der Datei ein, fragt die Antworten vom Benutzer ab und vergleicht sie mit den korrekten Antworten. Am Ende wird dem Benutzer eine Auswertung präsentiert.

1.2 Aufbau der Datei

Die Fragen sind zusammen mit den korrekten Antworten in einer Datei binär gespeichert. In den ersten vier Byte steht die Anzahl der Fragen im *int*-Format. Dahinter sind die entsprechende Anzahl an Fragen gespeichert, die jeweils aus einem Fragetext und der korrekten Antwort bestehen.

Für eine Frage steht in der Datei zunächst die Länge der Zeichenkette als eine vier Byte *int*-Variable (ohne die in C benötigte abschließende null). Dahinter folgt eine Kette von Zeichen entsprechend der angegebenen Länge. Schließlich wird für jede Frage die korrekte Antwort im vier Byte *int*-Format angegeben.

Die folgende Tabelle fasst den Aufbau der Datei noch einmal zusammen:

Anzahl	Größe	Datentyp	Inhalt
einmal	4 Byte	**int**	Anzahl der Fragen n
n-mal	4 Byte	**int**	Anzahl der Zeichen im Fragetext m
	m Byte	**char**[ ]	Fragetext (ohne abschließende null)
	4 Byte	**int**	richtige Antwort

2 Programmieraufgaben

Aufgabe 1 (10 Punkte)

Erstellen Sie für die Fragen eine Struktur und definieren Sie dafür einen Typ mit Namen *tQuestion*. In die Struktur soll dynamisch der Fragetext, statisch die Antwort des Benutzers

und wieder statisch die korrekte Antwort gespeichert werden. Für die Sammlung aller Fragen im Quiz wird später ein dynamischer Vektor dieses Typs verwendet.

Aufgabe 2 (20 Punkte)

Erstellen Sie eine Funktion mit Namen *readQuiz* zum Einlesen aller Fragen aus der oben beschriebenen Binärdatei. Die Funktion erwartet als Parameter den Dateinamen und einen Zeiger auf *int*, über den die Funktion die Anzahl der Fragen zurückgibt.

Die Fragen (mit den Antworten) werden dynamisch als Vektor gespeichert. Für die Fragetexte wird in Abhängigkeit der Textlänge ebenfalls dynamisch Speicher reserviert.

Bei Erfolg liefert die Funktion die Adresse des dynamisch reservierten Vektors, bei Misserfolg die Adresse *NULL* zurück.

Aufgabe 3 (10 Punkte)

Erstellen Sie eine Funktion mit Namen *getAnswers*, die alle eingelesenen Fragen dem Benutzer stellt und die dazu gehörigen Antworten abfragt. Die Ergebnisse werden in die Struktur der Fragen eingetragen.

Aufgabe 4 (10 Punkte)

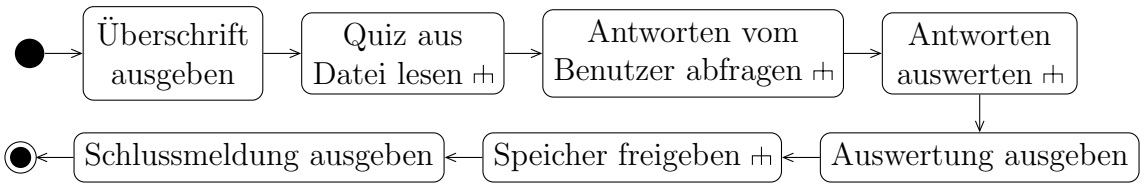
Erstellen Sie eine Funktion mit Namen *getResult* zum Auswerten der Antworten. Die Funktion gibt für jede Frage an, ob die Antwort korrekt oder falsch war. Als Rückgabewert gibt die Funktion die Anzahl der korrekt beantworteten Fragen zurück.

Aufgabe 5 (8 Punkte)

Erstellen Sie eine Funktion mit Namen *freeQuiz*, die den dynamisch reservierten Speicher wieder freigibt.

Aufgabe 6 (12 Punkte)

Fügen Sie die Programmstücke gemäß folgendem Aktivitätsdiagramm zusammen:



Achten Sie auf eine gute Benutzerführung, fangen Sie mögliche Fehleingaben des Benutzers ab, vermeiden Sie globale Variablen und entfernen Sie alle Fehler und Warnungen.

3 Schriftliche Aufgaben

Aufgabe 7 (3 Punkte)

Erklären Sie den Unterschied zwischen den Befehlen **#include** <stdio.h> und **#include** "stdio.h".

Nachname: ..... Vorname: ..... MatrNo.: .....

Aufgabe 8 (3 Punkte)

Erläutern Sie den Begriff *Traversierung* im Kontext von Rekursion.

Aufgabe 9 (6 Punkte)

a) Definieren Sie einen Zeiger auf Zeiger auf *double* und initialsieren Sie ihn mit NULL.

.....

b) Definieren Sie einen Vektor mit vier *unsigned* variablen und initialiseren Sie ihn mit den Werten 1, 2, 3 und 4.

.....

c) Bestimmen Sie die Größe im Speicher auf unseren Systemen in Byte.

Variable a): ..... Variable b): .....

Aufgabe 10 (3 Punkte)

Sie wollen alle 100 Elemente eines Vektors mit null initialisieren. Welche Schleife eignet sich am besten?

☐ *for*-Schleife ☐ *do*-Schleife ☐ *while*-Schleife

Aufgabe 11 (15 Punkte)

Erstellen Sie ein Aktivitätsdiagramm für die folgende Funktion:

```
unsigned gcd(int a, int b)
{
    unsigned x1;    // larger value
    unsigned x2;    // smaller value
    unsigned tmp;   // temporary vlaue to swap variables

    x1 = a>b?a:b;   // copy larger parameter
    x2 = a>b?b:a;   // copy smaller parameter

    // repeat until no remainder
    while(x1%x2) {
        tmp = x2;   // memorize previous smaller value
        x2 = x1%x2; // evaluate new smaller value
        x1 = tmp;   // use previous smaller value as new larger value
    }

    // return greatest common divisor
    return x2;
}
```