FAKULTÄT FÜR
WIRTSCHAFTSWISSENSCHAFT

LEHRSTUHL FÜR
BETRIEBSWIRTSCHAFTSLEHRE, INSBES.
MANAGEMENT SCIENCE

**PD Dr. Andreas Bortfeldt**

**Klausur zur Lehrveranstaltung**

**Programmieren in C++ (20757)**
**15. Juli 2013**

Name:..... Matrikelnummer:..... .....

Allgemeine Hinweise:

1. Schreiben Sie nach dem Ausfüllen dieses Deckblattes nochmals auf alle Ihnen ausgehändigten Blätter Ihren Namen und Ihre Matrikelnummer!
2. Kontrollieren Sie vor Beginn der Bearbeitung der Klausur die Vollständigkeit der Klausur! Die Klausur umfasst **8 Aufgaben**, von denen alle zu bearbeiten sind. Das Lösen der Heftklammern ist nicht gestattet und wird als Täuschungsversuch geahndet.
3. Schreiben Sie leserlich! Verwenden Sie nur Schreibgeräte mit dokumentenechter Tinte! Die Verwendung von Bleistiften oder roter Tinte ist nicht zugelassen.
4. Schreiben Sie zu jeder Aufgabe Ihre Lösung an den vorgesehenen Platz. Die Rückseiten der Blätter können für Notizen genutzt werden; diese werden jedoch nicht bewertet.
5. Erlaubte Hilfsmittel: Schreibgeräte, nicht-programmierbare Taschenrechner ohne Kommunikations- oder Textverarbeitungsfunktion, Vorlesungsskript einschließlich Quelltexte bereitgestellter Beispielprogramme, Wörterbücher.

**Aufgabe 1 (3 Punkte)**

Entscheiden Sie, ob folgende Literale (d.h. unbenannte Konstanten) im Wertebereich des jeweils angegebenen Datentyps liegen.

| Datentyp | Literal | Literal liegt im Wertebereich des Datentyps (ja oder nein) |
|---|---|---|
| unsigned char | '\t' | |
| double | 123123.999 | |
| unsigned int | 0 | |
| char | "x" | |
| int | 999.1 | |
| signed int | 150 | |

**Aufgabe 2 (3 Punkte)**

Welchen Wert haben die unten angegebenen Variablen nach den jeweils aufgeführten Anweisungen?

a)
```
int x = 2, y = 0, z = 0;
if (x  > 1 && y > 0 || z > 0)
  if ( x > 2 || y > 10)
        z = 4;
else  z = 3;
```
Wert von *z*: _____

b)
```
int u = 0, v = 1, w = 0;
if (u + v + w > 0 && u > v)
  u = 1;
else if (u + w > 2)
  u = 2;
else u = 3;
```
Wert von *u*: _____

**Aufgabe 3 (4 Punkte)**

Geben Sie an, wie oft die folgende while-Schleife durchlaufen wird.
Begründen Sie Ihre Antwort!

```
int x = 5, y = 2;
while (x + 1 > y)
{
        x = x/y;
        if (x == 1)
                x = x + 1;
}
```

Die while-Schleife wird _____ durchlaufen.

Begründung: _____

_____

_____

_____

**Aufgabe 4 (5 Punkte)**

Die Variablen *x* und *y* seien wie folgt definiert und initialisiert:

int x = 3, y = 6;

Welche Werte enthält die int-Variable *z* nach folgenden Zuweisungen?

a) z = 2*x/y; Wert von z: _____

b) z = (x - 2)/2; Wert von z: _____

c) z = x*3/4; Wert von z: _____

d) z = (int)(1. /2*y); Wert von z: _____

e) z = y / 2 / 4; Wert von z: _____

**Aufgabe 5 (5 Punkte)**

Schreiben Sie eine Funktion, mit der *wahlweise* ein Geldbetrag in Euro (EUR) in einen Betrag in Singapur-Dollar (SGD) umgerechnet werden kann oder ein Betrag in Singapur-Dollar in einen Euro-Betrag. Die Umrechnung soll mittels folgender Kurse erfolgen:

1 SGD = 0.61 EUR und 1 EUR = 1.64 SGD.

Die Funktion soll den Namen *change_sgd_eur(...)* besitzen. Steuern Sie die Variante der Umrechnung über einen Parameter, der nur die Werte 0 und 1 annehmen soll. Der umzurechnende Geldbetrag soll als zweiter Parameter der Funktion erscheinen, während der berechnete Geldbetrag als Returnwert von der Funktion zurückzugeben ist. Bei einem umzurechnenden Betrag kleiner oder gleich 0.00 Geldeinheiten ist die Berechnung mit einer Fehlermeldung und der Rückgabe des negativen Geldbetrags -1.0 zu beenden. Ergänzen Sie den fehlenden Code in dem folgenden Kasten!

```
                change_sgd_eur(                                                    )
{
   // Lokale Variablen



   // Anweisungen









}
```

**Aufgabe 6 (10 Punkte)**

a) Schreiben Sie eine Funktion check_password(...) gemäß folgender Spezifikation:

- Bei einem Aufruf der Funktion kann ein Password mit maximal 8 ASCII-Zeichen maximal dreimal von der Tastatur eingelesen werden.
- Bei jedem Eingabeversuch wird die eingelesene Zeichenkette mit dem korrekten Password verglichen. Das korrekte Password wird in der Funktion gespeichert und lautet "abraca#1".
- Zum Vergleich wird die Stringfunktion strcmp aus der Standard-Bibliothek genutzt. Diese Funktion wird für zwei Stringzeiger p1 und p2 gemäß strcmp(p1, p2) aufgerufen. Sie gibt den Wert 0 zurück, falls die beiden Strings p1 und p2 übereinstimmen und andernfalls einen von 0 verschiedenen int-Wert.
- Die Funktion bricht ab, nachdem das Password korrekt eingeben wurde oder nach dem dritten negativ verlaufenen Eingabeversuch. Im ersten Fall soll die Funktion den Wert 1, sonst den Wert 0 zurückgeben.
- Der Nutzer der Funktion soll zu jeder Eingabe am Bildschirm aufgefordert und über das Ergebnis jedes Versuchs unterrichtet werden.
- Es kann vorausgesetzt werden, dass ein Nutzer der Funktion nur Zeichenketten ohne Leerzeichen oder andere Zwischenraumzeichen eingibt.

Ergänzen Sie den fehlenden Code in dem folgenden Kasten!

b) Eine Modifikation der Funktion check_password(...) sieht vor, dass die Funktion das korrekte Password über einen Parameter erhält. Schreiben Sie hierzu einen geeigneten abgeänderten Funktionskopf auf und geben Sie den abgeänderten Aufruf der Funktion strcmp(...) an. Nutzen Sie für Ihre Antworten die vorgesehenen Stellen.

```
        check_password(                                )
{
   // lokale Variablen




   // Anweisungen









}
```

Modifizierter Funktionskopf:

_____

Modifizierter Aufruf von strcmp():

strcmp(_____)

**Aufgabe 7 (20 Punkte)**

Es ist eine Klasse *Weinkaraffe* gemäß folgender Spezifikation zu programmieren:

- Ein Objekt der Klasse bzw. eine Weinkaraffe soll zwei Datenelemente besitzen: zum einen die maximale Füllmenge (in Litern) und zum anderen die aktuelle Füllmenge (in Litern). Als Füllmengen kommen nur *ganzzahlige* Literbeträge in Betracht.
- Für die Klasse Weinkaraffe soll ein Konstruktor mit einem Parameter zur Verfügung stehen, welcher die maximale Füllmenge (einmalig) festlegt. Das zweite Datenelement ist ebenfalls geeignet zu initialisieren.
- Mit einem Objekt bzw. einer Weinkaraffe sollen folgende Operationen möglich sein:
  1) Leeren der Karaffe, d.h. die aktuelle Füllmenge wird auf null reduziert.
  2) Füllen der Karaffe, d.h. die aktuelle Füllmenge wird auf die maximale Füllmenge gesetzt.
  3) Ein Test, ob die Karaffe voll ist, d.h. die Operation soll den Wert 1 zurückgeben, wenn die aktuelle Füllmenge gleich der maximalen Füllmenge ist und 0 sonst.
  4) Ein Test, ob die Karaffe leer ist, d.h. die Operation soll den Wert 1 zurückgeben, wenn die aktuelle Füllmenge gleich null ist und 0 sonst.
  5) Auffüllen der Karaffe mit einem gewissen ganzzahligen Literbetrag, welcher zur aktuellen Füllmenge hinzukommt. Dabei ist auszuschließen, dass die maximale Füllmenge überschritten wird. Führt die hinzukommende Menge zu einer Überschreitung, so ist die Karaffe nur bis zur maximalen Füllmenge aufzufüllen.

a) Tragen Sie in dem folgenden Kasten (S. 8) Ihre Definition der Klasse Weinkaraffe ein. Treffen Sie dabei insbesondere eine geeignete Entscheidung im Hinblick auf die Sichtbarkeit der Datenelemente und Funktionen.

b) Füllen Sie die dann folgenden Kästen (S. 9 und 10) mit den Definitionen der laut Spezifikation vorgesehenen Funktionen der Klasse Weinkaraffe.

a) Definition der Klasse Weinkaraffe (für Headerdatei Weinkaraffe.h)

```
class
{
        public:

        private:

};
```

b) Definition der Funktionen (für Funktionsdatei Weinkaraffe.cpp)

b) Definition der Funktionen (für Funktionsdatei Weinkaraffe.cpp) (Fortsetzung)

**Aufgabe 8 (10 Punkte)**

In einer Klasse Geschäftspartner wird jeder Geschäftspartner eines Unternehmens mit den Datenelementen Firmenname, Name eines Ansprechpartners, E-Mail-Adresse und bisher erzielter Umsatz mit dem Geschäftspartner (in Euro) geführt. Es folgt die (unvollständige) Klassendefinition der Klasse Geschäftspartner.

```
class Geschäftspartner
{

        public:
                Geschäftspartner (char *p_firma, char *p_aname, char *p_email,
                                  double  bish_umsatz);

                void setFirma (char *p_firma);
                char *GetFirma (void);
                // weitere Zugriffsfunktionen

        private:
                char        firma[101];          // Firmenname
                char        ansprechpartner[51];// Name Ansprechpartner
                char        email[101];          // E-Mail-Adresse
                double      umsatz;              // bisher erzielter Umsatz

};
```

Die Klasse Geschäftspartner wird abgeleitet und erhält die Unterklasse Kunde. Die Klasse Kunde soll zusätzlich zur Klasse Geschäftspartner lediglich das Datenelement bonität vom Typ int enthalten, welches mit der Sichtbarkeit private zu spezifizieren ist. Ferner einen Konstruktor, mit welchem alle Datenelemente der abgeleiteten Klasse wie auch der Oberklasse initialisiert werden können sowie Zugriffsfunktionen für das Datenelement bonität.

a) Ergänzen Sie die im folgenden Kasten angegebene Klassendefinition der Klasse Kunde entsprechend den zuvor aufgelisteten Anforderungen.

b) Definieren Sie in dem dann folgenden Kasten den Konstruktor der Klasse Kunde. Beachten Sie dabei, dass mit einem Basisinitialisierer dafür gesorgt werden muss, dass auch das Teilobjekt der Klasse Geschäftspartner geeignet initialisiert wird.

a) Klassendefinition der Klasse Kunde (für Datei Kunde.h)

```
class Kunde : public Geschäftspartner
{

        public:         // zu ergänzen









        private:        // zu ergänzen









};
```

b) Definition des Konstruktors der Klasse Kunde (für Datei Kunde.cpp)