C++ Nachklausur WS 2008/09

Aufgabe 1: Fakultäten **(6 Punkte)**

Eine C++-Funktion soll die Fakultät $n!$ einer Zahl n berechnen und zurückgeben. (Zur Erinnerung: $n! = 1 \cdot 2 \cdot \ldots \cdot n$).
Schreiben Sie zwei voneinander unabhängige Varianten der Funktion: (a) iterativ, (b) rekursiv. Sie brauchen nicht zu überprüfen, ob das Funktionsargument positiv ist.

Aufgabe 2: Skalarprodukt **(3 Punkte)**

Schreiben Sie eine C++-Funktion `skaprod` zur Berechnung des Skalarprodukts von zwei n-dimensionalen Vektoren. Die Funktion soll als Argumente zwei Felder aus `double` Variablen und die Länge n der Felder erhalten. Gültige Feldindizes sind im Bereich $0, \ldots, n-1$.
Der Rückgabewert der Funktion soll das Skalarprodukt sein.

Aufgabe 3: Minuten und Sekunden **(7 Punkte)**

Es ist eine Klasse `minsek` zu programmieren, die zur Aufbewahrung und zum Rechnen von Zeitabschnitten dient. Die Zeitabschnitte sollen jeweils durch Minuten- und Sekundenbetrag gekennzeichnet sein. Realisieren Sie diese Werte durch von außen unzugängliche `int`-Variablen. Die Klasse soll in folgendem Programm zum Einsatz kommen:

```
#include <iostream>
using namespace std ;

// Klasse minsek zu ergaenzen

int main()
{
    minsek triathlon(0,0) ;      // ein Nano-Triathlon:
    triathlon += minsek(2,41) ; // Schwimmen, 2:41
    triathlon += minsek(1,33) ; // Radfahren, 1:33
    triathlon += minsek(3,27) ; // Laufen, 3:27

    cout << "Gesamtzeit: " << get_min(triathlon) << " Minuten "
         << " und " << get_sek(triathlon) << " Sekunden" << endl ;
}
```

Schreiben Sie die benötigte Klasse `minsek` und alle benötigten Konstruktoren, Mitgliedsfunktionen und freien Funktionen, die in dem gegebenen Hauptprogramm aufgerufen werden. Sie dürfen davon ausgehen, dass dem Konstruktor nur übliche Variablenbereiche (Sekunden zwischen 0 und 59) übergeben werden. Bei der Addition von Zeiten soll der Überlauf aber richtig behandelt werden.

Aufgabe 4: Was wird ausgegeben? (5 Punkte)

Setzen Sie an der markierten Stelle in folgendem Programm die letzte Ziffer Ihrer Matrikelnummer ein. Schreiben Sie dann *vollständig* auf, was von dem Programm ausgegeben wird.

```
#include <iostream>
using namespace std ;

void f(int p, int q) {
    q+=2 ;
      cout << "Fkt1: " << q << endl ;
    if (p>0) {
      f(p-1,q) ;
    }
    return ;
}

void f(int p, double& q) {
    q += p ;
    cout << "Fkt2: " << q << endl ;
    return ;
}

int main() {
    int a1= _ ; // <- setzen Sie hier bitte die letzte Ziffer
                //    Ihrer Matrikelnummer ein.
    double a2 = a1 + 0.2 ;

    f(1,a2) ;
    cout << "Main1: " << a2 << endl ;
    f(1,a1) ;
    cout << "Main2: " << a1 << endl ;
}
```

Aufgabe 5: Dynamische Felder (3 Punkte)

Schreiben Sie die Befehlszeilen für Folgendes auf:
(a) Allokierung eines dynamischen Feldes von n `double` Variablen. Deklarieren Sie dabei auch die Feldvariable, den Namen des Feldes können Sie selbst bestimmen.
(b) Freigabe des Feldspeichers (Feld aus Teil (a)).

Aufgabe 6: Zerlegung in zwei Primzahlen (6 Punkte)

Die Goldbachsche Vermutung lautet: *Jede gerade Zahl größer als 2 kann als Summe zweier Primzahlen geschrieben werden.* Diese Vermutung ist bis heute unbewiesen.
Schreiben Sie ein Programm, welches Folgendes leistet: Zu einer eingegebenen Zahl sollen alle möglichen Zerlegungen der Zahl in zwei Primzahlsummanden ermittelt und ausgegeben werden. Sie brauchen nicht zu überprüfen, ob die eingegebene Zahl größer als 2 und gerade ist.
Bsp: Eingabe: 14 $\longrightarrow$ Ausgabe: 3+11 7+7.
Hinweise: 1 ist keine Primzahl. Durch Programmierung geeigneter Funktionen lässt sich die Aufgabe in Teilprobleme zerlegen.