(4 Punkte)

Aufgabe 1: Dreieckstest

Schreiben Sie eine Funktion `dreieckstest`, die drei `double`-Argumente hat und einen Logikwert als Ergebnis zurückgibt. Die Funktion soll ermitteln, ob die drei Argumente als Seitenlängen eines Dreiecks aufgefasst werden können und genau in diesem Fall `true` zum Ergebnis haben.
Hinweis für die Testkriterien: Bei einem Dreieck müssen die Seitenlängen echt positiv sein und jeweils zwei beliebige Seiten müssen zusammen länger als die dritte Seite sein.

(5 Punkte)

Aufgabe 2: Zeilen in Matrix tauschen

Schreiben Sie eine Funktion `ztausch`, die zwei Zeilen in einer $n \times n$ Matrix austauscht. Der Aufruf der Funktion ist folgendermaßen: `ztausch( a, n, i, j)`. Dabei steht `a` für ein eindimensionales Feld der Länge `n*n`, in dem Elemente der Matrix zeilenweise abgelegt sind. Die Argumente `i` und `j` bezeichnen die beiden zu vertauschenden Zeilen.
Sie dürfen voraussetzen, dass $n \in \mathbb{N}$, $i, j \in \mathbb{N}_0$ und $0 \leq i, j < n$ gilt.

(7 Punkte)

Aufgabe 3: Geraden-Klasse

Schreiben Sie eine Klasse `gerade`, die zur Beschreibung von Geraden, also Ausdrücken der Form $a \cdot x + b$ dient. Geraden werden durch die beiden Parameter a und b charakterisiert, diese sollen von außerhalb der Klasse nicht direkt zugänglich sein. Die Steigung a soll ganzzahlig sein, der Achsenabschnitt b eine Gleitkommazahl.
Programmieren Sie folgende Methoden und Funktionen, so dass untenstehendes Programm funktioniert: (i) Konstruktoren; (ii) Berechnung des y-Werts für einen x-Wert; (iii) Addition zweier Geradengleichungen; (iv) Test, ob zwei Geraden genau einen gemeinsamen Schnittpunkt haben.
Die Klasse kommt in folgendem Programm zum Einsatz:

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

// Klasse gerade und freie Funktion zu ergaenzen

int main()
{
    gerade g1 ( -1, 2.5) ;  // Gerade 1: y =  -x + 2.5 ;
    gerade g2 (  2, -0.5) ; // Gerade 2: y = 2*x - 0.5 ;

                            // Gerade 3, ohne initialisierende Werte
    gerade g3 ;
    g3 = g1 + g2 ;

    cout << "g3(4.2): " << g3.at(4.2)  << endl ; //  g3 für x=4.2 ausgewertet

    if (genau_ein_schnittpunkt(g1, g2))
       cout << "g1 und g2 haben genau einen Schnittpunkt" << endl ;

}
```

Schreiben Sie die benötigte Klasse `gerade` und alle benötigten Konstruktoren, Mitgliedsfunktionen und freien Funktionen, die in dem gegebenen Hauptprogramm aufgerufen werden.
Der Rückgabewert von `genau_ein_schnittpunkt` soll ein Logikwert sein.

Aufgabe 4: Einfach verkettete Liste (7 Punkte)

```
struct listelem {
  double value;
  listelem* next;
};
```

Mit obiger Struktur seien einfach-verkettete Listen aufgebaut. Sie dürfen davon ausgehen, dass stets gültige Zeiger auf Listenelemente an die zu erstellenden Funktionen übergeben werden und dass das Ende einer Listenkette mit dem Nullzeiger markiert ist.

(a) Schreiben Sie ein Funktion `addfront`, die als Argument einen Zeiger auf ein Listenelement erhält und einen Zeiger auf ein Listenelement zurückgibt. Die Funktion soll: (i) dynamisch ein neues Listenelement erzeugen; (ii) den Wert des neuen Elements auf 2016.7 setzen; (iii) an das neue Element die Liste des Funktionsarguments anhängen; (iv) einen Zeiger auf das neue Element zurückgeben.

(b) Schreiben Sie eine rein rekursive Funktion `listsum`, die einen Zeiger auf ein Listenelement erhält und die Summe der `value`-Werte aller Listenelemente ab diesem Knoten aufaddiert und zurückgibt.

Aufgabe 5: Giuga-Zahlen (7 Punkte)

Als Giuga-Zahlen werden diejenigen Zahlen $n \in \mathbb{N}$ bezeichnet, die mindestens zwei verschiedene Primfaktoren haben und die die Eigenschaft haben, dass jeder Primfaktor p von n auch den Ausdruck $n/p - 1$ restlos teilt.
Schreiben Sie eine C++-Funktion `is_giuga`, die ermittelt, ob eine Zahl n eine Giuga-Zahl ist. Die Funktion soll per Logikwert das Ergebnis zurückgeben.

Zur Lösung wird die Zerlegung der Aufgabenstellung in geeignete Funktionen empfohlen.
Kommentieren Sie die einzelnen Abschnitte in Ihrem Programm, so dass sofort verständlich ist, was jeweils gemacht wird. Ansonsten kann der entsprechende Programmteil nicht bewertet werden.

Bsp: 30 ist eine Giuga-Zahl, da $30 = 2 \cdot 3 \cdot 5$ und die folgenden Teilbarkeiten erfüllt sind:
$2 \mid (30/2 - 1)$, $3 \mid (30/3 - 1)$ und $5 \mid (30/5 - 1)$.

Hinweise: Der Operator % ergibt den Rest einer Ganzzahl-Division.
Die Zahl 1 ist keine Primzahl.